Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

ZAVRŠNI RAD

Tomislav Radić

Zagreb, 2012.

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

ZAVRŠNI RAD

Voditelj rada:

Prof. dr. sc. Bojan Jerbić

Tomislav Radić

Zagreb, 2012.

Izjavljujem da sam ovaj rad izradio samostalno, služeći se znanjem stečenim tijekom studija i koristeći navedenu literaturu.

Ovom prilikom bih želio zahvaliti:

Voditelju rada prof. dr.sc. Bojanu Jerbić te asistentu mag.ing Bojanu Šekoranja na stručnim savjetima i pomoći tijekom izrade završnog rada.

Posebno bih želio zahvaliti svojoj obitelji - roditeljima Stjepanu i Nadi te bratu Vedranu na razumijevanju, potpori i pomoći kako tijekom izrade ovog rada, tako i tijekom cijelog studija.

Također zahvaljujem svima ostalima koji su mi na bilo koji način pomogli oko ovog rada.

SADRŽAJ:

POPIS SLIKA

POPIS TABLICA

POPIS KRATICA I STRANIH RIJEĆI

TCP/IP - Transmission Control Protocol/Internet Protocol

WiFi = WLAN - wireless local area network

LCD - liquid crystal display

IPS - In-Plane Switching

LED - light-emitting diode

USB - Universal Serial Bus

EDR - Enhanced Data Rate

SDK - software development kit

Home - doma

Sleep - spavaj

Volume - glasnoća

Software - softver

Standby – stanje pripravnosti

Finish - završi

Source - izvorno

Design - dizajn

Components - komponente

Button - gumb

Hslider – horizontalni klizač

Label - oznaka

Image - slika

TextInput – unos teksta

Run Configurations – pokreni konfigurecije

On desktop – na zaslon

Socket - utor

Receive data – primljeni podatci

Send data – poslani podatci

Project - projekt

Export Release Build – finalno izdanje

Cetificate - certifikat

Provisioning file – datoteka odobrenja

Handling Tool – rukovanje alatom

Motion - gibanje

Motion grups – grupe gibanja

Socket messaging – slanje poruka preko utičnice

Yes - da

Build - izgradi

Robot Neighborhood – robotsko susjedstvo

Export – izvoz

pick & place –pokupi i postavi

Multi Touch – više dodirnih točaka s površinom

1. Uvod

Cilj ovog rada bio je razvit komunikacijske programe za bežičnu komunikaciju robotske upravljačke jedinice i iPad uređaja. Na strani iPad uređaja izrađena je aplikacija koja putem TCP/IP protokola ima mogućnost prenositi naredbe korisnika prema upravljačkoj jedinici robota. Za upravljačku jedinicu robota izrađen je program koji interpretira primljene naredbe i šalje povratne informacije o radu robota u realnom vremenu. Povratna veza omogućuje aplikaciji da na zaslonu iPad uređaja prikazuje položaj robotske ruke u prostoru.

Program na strani robotske upravljačke jedinice je izrađen u Roboguide simulacijskom paketu za *offline* programiranje Fanuc robota, te korištenjem Karel programskog jezika. Glavni dio zadatka, aplikacija za iPad uređaj izrađena je u Flash Builder 4.6 programu. Flash Builder je Adobe-ov razvojni alat za izradu mobilnih, web i desktop aplikacija. Koristi ActionScript programski jezik i Flex *open source* radno okruženje. Zadatak je implementiran na robotu FanucLRM200iC/5L u Labaratoriju za projektiranje izradbenih i montažnih sustava.

2. Apple iPad uređaj

IPad je linija tablet računala dizajnirana i plasirana na tržište od strane američke računalne tvrtke Apple Inc. Demonstrira potencijal tableta kao segmenta i s vremenom, uz novije verzije uređaja sa softverskim i hardverskim poboljšanjima polako opravdava pompu koju je doživljavao od pojavljivanja na tržištu. Prva generacija je puštena u prodaju u travnju, 2010. godine i do danas je prodano više od 67 milijuna primjeraka.

Prvenstveno služi kao platforma za reproduciranje audio-vizualnih medija, uključujući knjige, časopise, filmove, glazbu, igre, aplikacije i web sadržaje. IPad-om se upravlja preko 9.7 inčnog zaslona osjetljivog na dodir i dostupan je u varijantama od 16, 32 i 64 gigabajta memorijskog prostora. Koristi WiFi tehnologiju za pristup lokalnim mrežama i internetu.

Za instalaciju i pokretanje aplikacije izrađene u ovom završnom radu korišten je iPad uređaj prve generacije.

Slika 1. IPad prva generacija

2.1. Hardver

2.1.1. Zaslon i ulazi

IPad-ov zaslon je dijagonale 9.7 inča i rezolucije 1024 x 768 piksela. Koristi vrhunsku tehnologiju LCD ekrana, IPS. Ova tehnologija pruža izvrsne kuteve gledanja i duboke boje. U kombinaciji sa pozadinskim LED osvjetljenjem zaslon vrlo dobro utjelovljuje sve što se reproducira na njemu. Također je vrlo osjetljiv i precizan za kontrolu dodirom.

Senzor ambijentalnog osvjetljenja podešava svjetlinu zaslona a tri-osni akcelerometar prebacuje način pogleda iz portreta u krajolik i obrnuto ovisno o orjentaciji, tako da uređaj nema svoju prirodnu orjentaciju nego samo relativnu poziciju *home* tipke.

Slika 2. *Home* tipka

Na uređaju postoje samo četiri prekidača, uključujući *home* tipku u blizini ekrana koja vraća korisnika na glavni izbornik, te tri na stranama: *sleep*, *volume* i *software* prekidač čija se funkcija mjenja softverski.

Slika 3. *Volume, Software , Sleep/Wake-On/Off* tipke

2.1.2. Povezivanje

Za pristup lokalnim mrežama i internetu iPad ima ugrađen Wi-Fi (802.11 a/b/g/n) mrežni adapter. Portova za USB i Ethernet konektore nema. Povezivanje sa Mac ili Windows PC-om moguće je samo Apple dock konektorom.

Slika 4. Kabal za povezivanje Ipad uređaja s PC-om

2.1.3.Zvuk i izlazi

IPad ima dva unutranja zvučnika za reprodukciju lijevog i desnog audio kanala. 3.5 mm *jack* priključak audio-izlaza u gornjem lijevom kutu uređaja daje stereo zvuk za slušalice sa ili bez mikrofona. Također uređaj sadrži mikrofon koji se može koristiti za snimanje glasa. Ugrađeno Bluetooth 2.1+EDR sučelje omogučuje korištenje bežičnih slušalica i tipkovnice. Operacijski sustav iOS trenutno ne podržava prijenos datoteka koristeći Bluetooth vezu.

2.1.4. Baterija

Baterija je litij-ionska polimerna, punjiva i ugrađena u sam uređaj te ju se ne može vaditi. Omogućuje reprodukciju videa do 10h, zvuka do 140h te u *standby* načinu može raditi i do mjesec dana.

2.1.5.Memorija

IPad dostupan je u varijantama od 16, 32 i 64 gigabajta memorijskog prostora. Svi se podatci pohranjuju na internu flash memoriju, bez mogućnosti širenja prostora za pohranu.

2.2. Softver

Uređaj radi na iOS operacijskom sustavu i može pokretati samo programe odabrane od strane Apple-a i distribuirane preko Apple App Store-a, te aplikacije napisane od razvojnih programera koji su platili razvojnu licencu. Razvojni programeri koriste iPhone SDK za razvoj aplikacija.

Poput i drugih iOS uređaja, iPad uređaj možemo otključati, čime aplikacije i programe koji nisu ovlašteni od strane Apple-a možemo pokrenuti na uređaju. Isto tako postoje različiti razvojni programi u kojima možemo izraditi takve aplikacije. Aplikacija za ovaj završni rad izrađena je u Adobe-ovom razvojnom programu FlashBuilderu 4.6 s podrškom za iOS operacijske sustave.

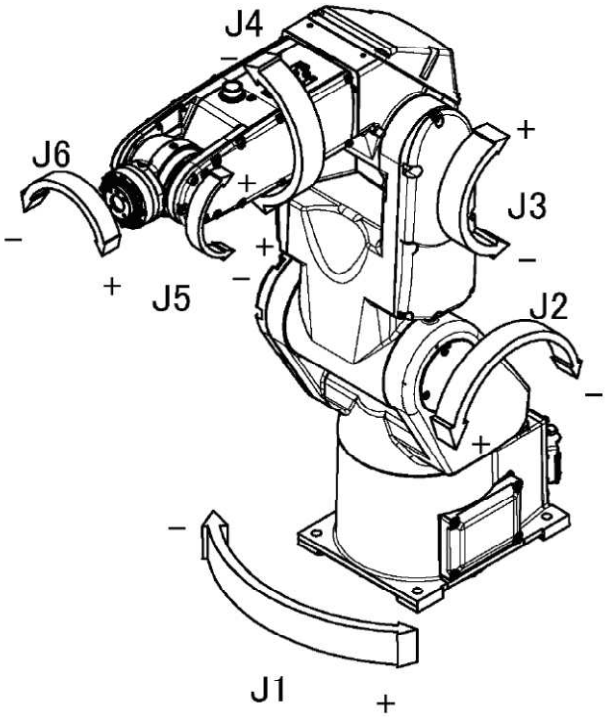
3. Fanuc LRM200iC/5L

LR Mate 200iC je električni servo upravljani mini robot proizvođača Fanuc Robotics. Nudi najbolje performanse u klasi. Lagan je, brz, efikasan i točan. Vitak profil ruke, manja težina, visoka spretnost, brža kontinuirana brzina i superiorna točnost pozicioniranja čine ga savršenim rješenjem za brojne industrijske i komercijalne operacije. Idealno je rješenje za rukovanje materijalom, sastavljanje, montiranje, kupljenje, pakiranje, ispitivanje. Također za obrazovanje i zabavu. Kao takav izabran je da na njemu bude testirana aplikacija za upravljanje robotom pomoću iPad uređaja.

Slika 5. FANUC LR Mate 200iC

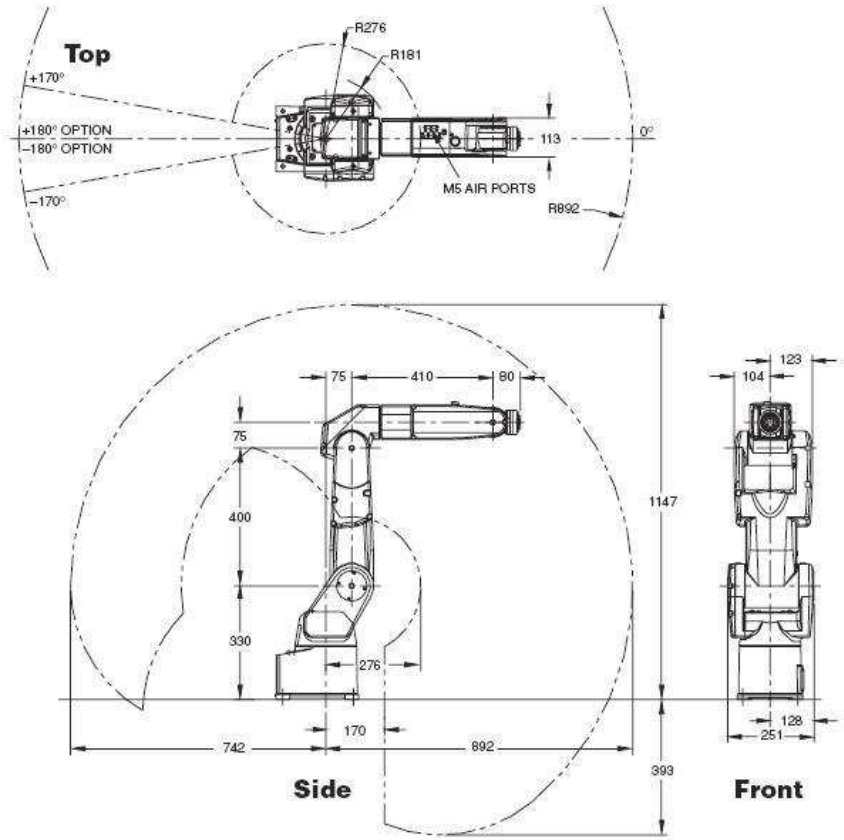
3.1. Tehničke karakteristike robota

Šest-osni LR Mate 200iC može na svom zapešću nositi teret do 5kg. Stolna veličina, tanki zglobovi te malo postolje mu dozovljavaju rad u skučenim radnim prostorima. Ima iznimnu ponovljivost od ±0.03mm pri punoj nosivosti i brzini unutar cijelog radnog prostora. Dizajnom robota i unutarnjom ugradnjom kablova i crijeva izbjegnuto je njihovo zapetljavanje.



Slika 6. Shematski prikaz šest-osnog upravljačkog sklopa (robota)–LR Mate 200*i*C5L

Ugradnja uspravno ili pod nekim kutem, na zid ili strop povećava fleksibilnost instalacije robota. Robot se može okrenuti unazad te je tako povećan opseg njegovog radnog prostora. Veća krutost i najnaprednija servo tehnologija omogućuju mu glatke pokrete bez vibracija pri velikim brzinama rada.



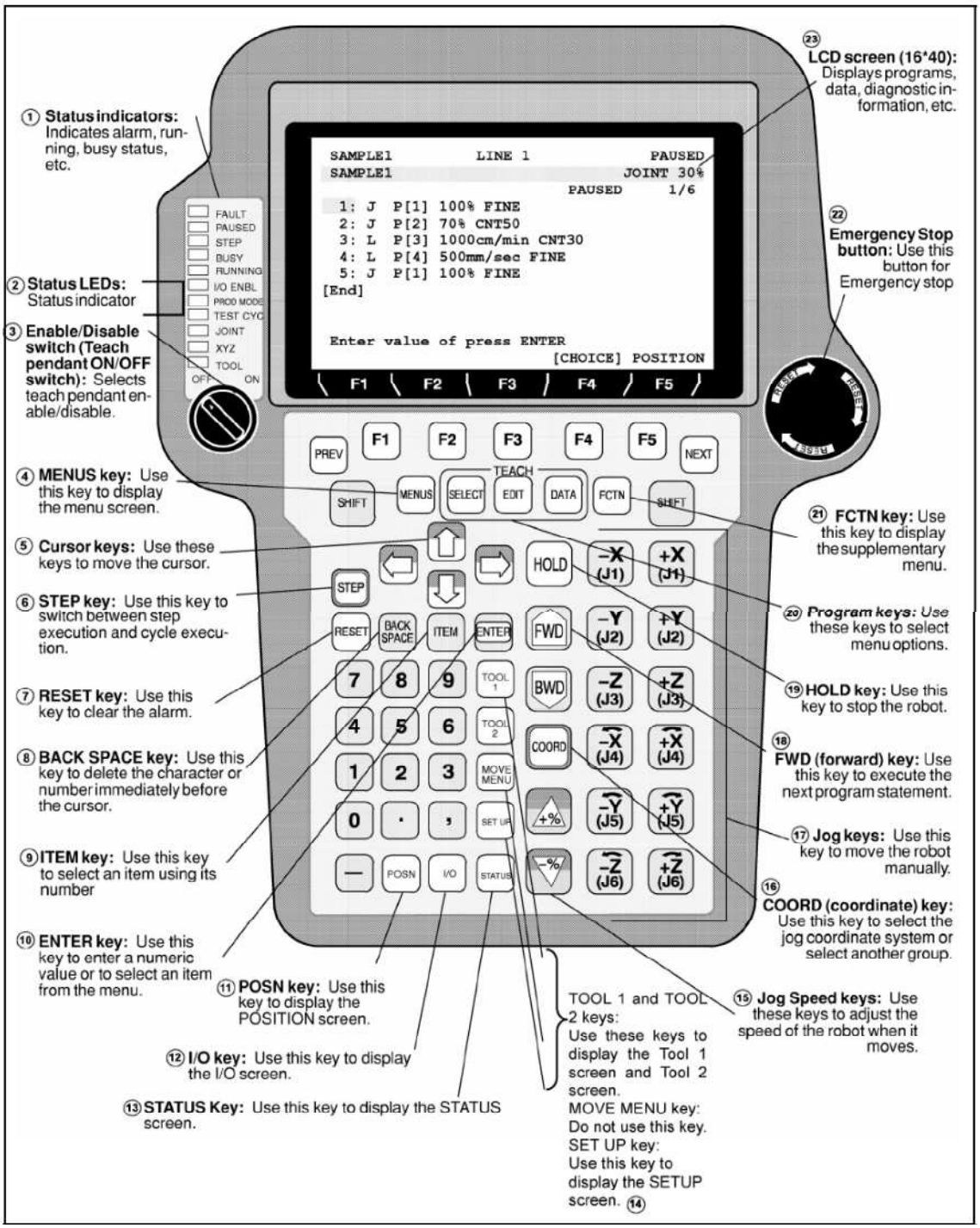
Slika 7. Dimenzje robota LR Mate 200iC/5L /5LC

Tablica 1. Osnovne tehničke karakteristike robotaFANUC LR Mate 200iC

Broj osi		6
Masa		29kg
Doseg		892mm
Točnost ponavljanja		±0.03mm
Maksimalna nisivost na zglobu		5kg
Opseg gibanja	J1	340° (5,93rad)
	J2	230° (4,01rad)
	J3	373° (6,51rad)
	J4	380° (6,63rad)
	J5	240° (4,19rad)
	J6	720° (12,57rad)
Maksimalna brzina	J1	270°/s (4,71rad/s)
	J2	270°/s (4,71rad/s)
	J3	270°/s (4,71rad/s)
	J4	450°/s (7,85rad/s)
	J5	450°/s (7,85rad/s)
	J6	720°/s (12,57rad/s)

3.2. Upravljačka konzola

Upravljačka konzola ili privjesak za učenje je ručni kontrolni terminal robota. Osigurava jednostavan način upravljanja robotom, izrade programa ili pokretanje već gotovih.



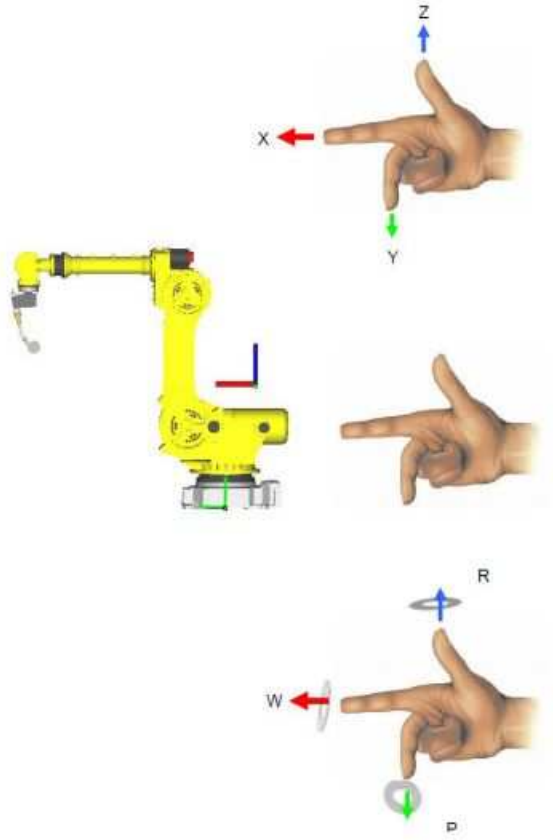
Slika 8. Prikaz upravljačke konzole sa pojašnjenjem namjene pojedinih tipki

Opis upravljačkih tipki :

1. Indikator stanja (eng. status indicator) – označava alarm, u radu, zauzet itd...
2. LED diode stanja – na upravljačkoj konzoli upravljačke jedinice R 30iA Mate ove oznake stanja nalaze se na zaslonu upravljačke konzole
3. Omogućena / onemogućena (eng. enable - disable) upravljačka konzola – ON/OFF sklopka
4. Tipka izbornik (eng. menu) – služi za prikazivanje glavnog izbornika
5. Tipke pokazivača (eng. cursor) – služe za pomicanje pokazivača
6. Tipka korak (eng. step) – služi za promjenu načina rada između koračnog i kontinuiranog načina izvršavanja naredbi
7. Tipka za vraćanje na izvorne postavke (eng. reset)
8. Tipka pomak unatrag (eng. backspace) – koristi se za brisanje broja ili znaka koji se nalazi prije pokazivača
9. Tipka predmet (eng. item) – koristi se za označavanje predmeta koristeći njegov broj
10. Tipka unos (eng. enter) – koristi se za upis numeričke vrijednosti ili znaka, označavanje programa, potvrde predmeta sa izbornika, itd...
11. Tipka pozicije (POSN, eng. position) – koristi se za prikaz zaslona pozicija robota
12. Tipka ulaz/izlaz (I/O, eng. input/output) – koristi se za prikaz zaslona sa ulazno izlaznim signalima robota
13. Tipka stanja (eng. status) – Koristi se za prikaz zaslona sa stanjem robota
14. Tipke alata (eng. tool) – prikaz zaslona alata
15. Tipka brzine kretanja robota (eng. jog speed) – koriste se za određivanje brzine robota prilikom izvođenja naredbi gibanja
16. Tipka koordinatnih sustava (COORD, eng. coordinate) – koristi se za izbor koordinatnog sustava robota
17. Tipka za ručno pomicanje robota (eng. Jog) – ručno pomicanje robota
18. Tipka naprijed (FWD, eng. forward) –koristi se za izvođenje slijedeće naredbe u programu
19. Tipka čekanja (eng. hold) – koristi se za zaustavljanje robota
20. Tipke programa (eng. program keys) – koriste se za izbor opcija izbornika
21. Tipka funkcije (FCTN) – prikaz dodatnog izbornika
22. Tipka hitnog zaustavljanja (eng. emergency stop button) – koristi se za trenutno zaustavljanje rada robota
23. LCD ekran upravljačke konzole

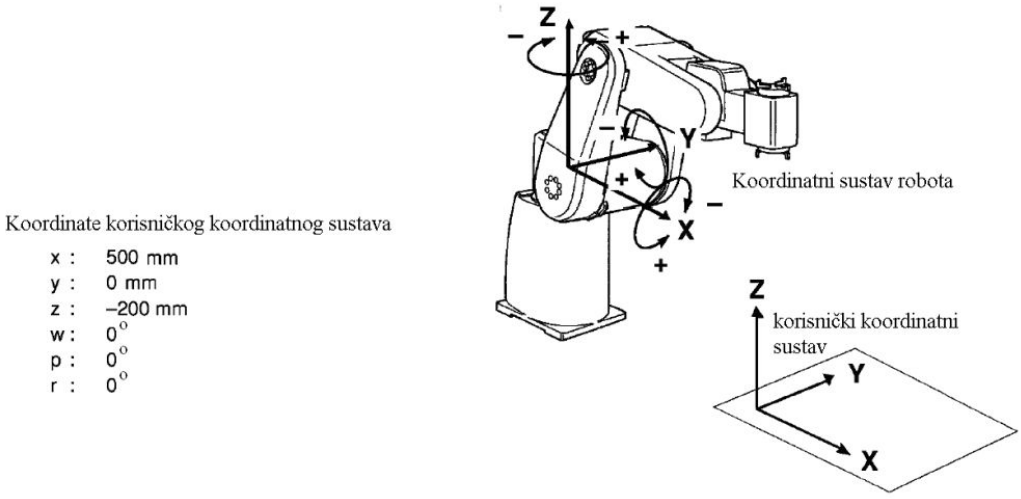
3.3. Koordinatni sustav robota

Pravokutni koordinatni sustav možemo u prostoru jednostavno predočiti pravilom desne ruke prikazanim na sljedećoj slici.



Slika 9. Shema koordinatnih sustava robota

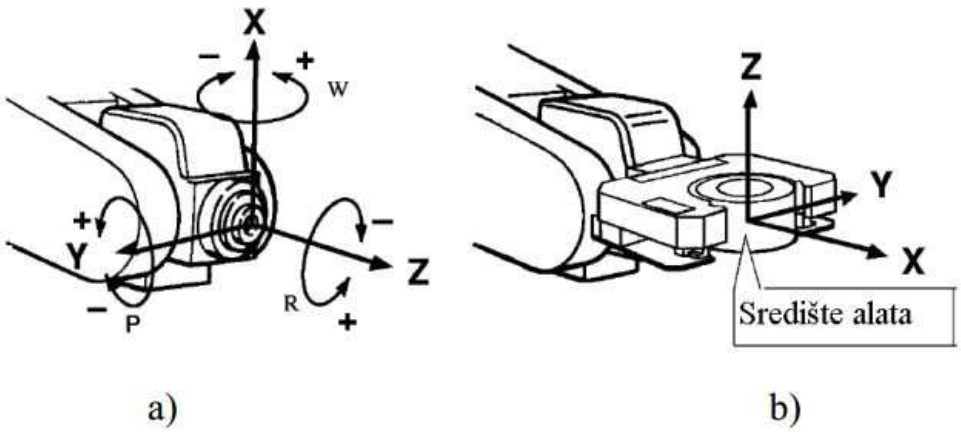
Po standardnim postavkama koordinatni sustav robota preddefiniran je u određenoj točci tijela robota. Ovaj koordinatni sustav je kartezijski, pravokutni, desnokretni koordinatni sustav. Svaki korisnički koordinatni sustav definira se u odnosu na ovaj osnovni koordinatni sustav i to prostornom translacijom po tri osi - x, y i z; te redom rotacijama oko tih triju osi – w, p i r. Moguće je definirati do devet korisničkih koordinatnih sustava. Prikaz koordinatnih sustava prikazan je sljedećom slikom.



Slika10. Koordinatni sustav robota, korisnički koordinatni sustav

3.4. Koordinatni sustav alata

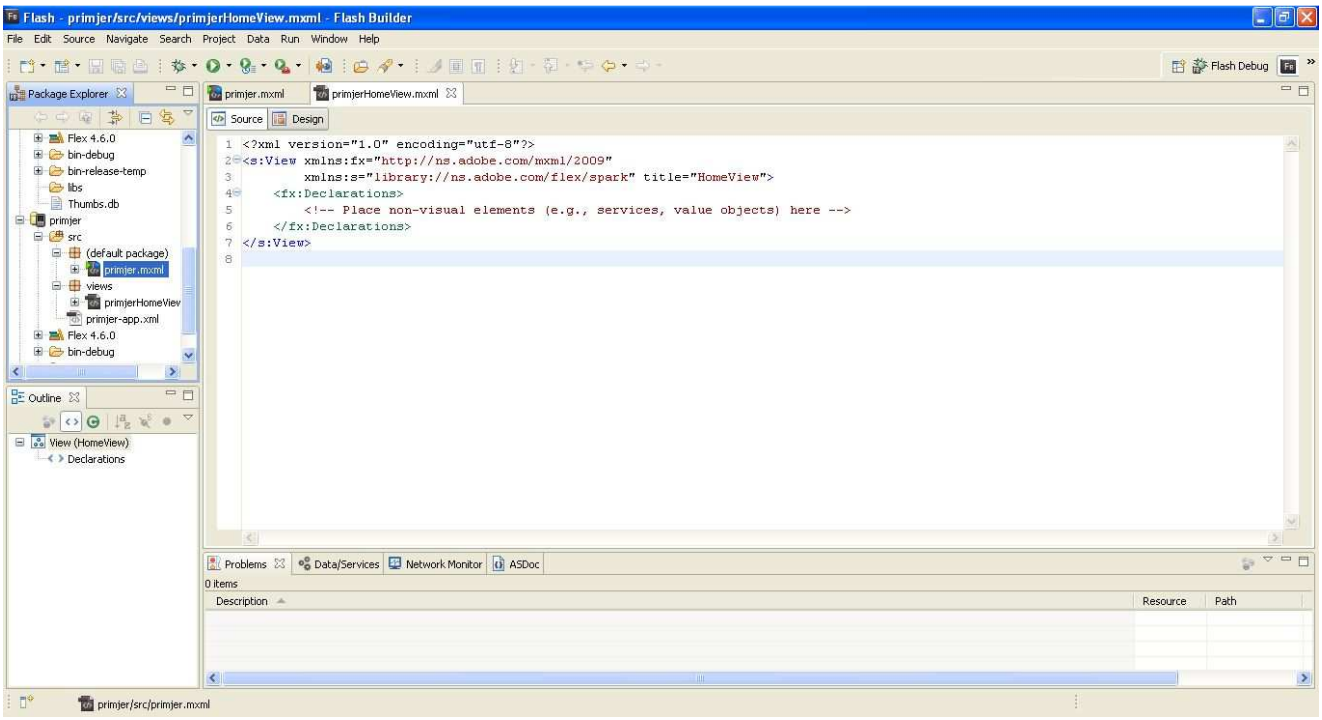
Ovisno o alatu kojeg robot koristi (npr. različite hvataljke) mora se definirati koordinatni sustav alata kako bi vrh alata uvijek izvršavao željeno gibanje. Analogno preddefiniranom koordinatnom sustavu svijeta robota postoji preddefiniran koordinatni sustav prihvata alata. On se nalazi kako je prikazano na slici u osi šestog zgloba. Prema ovoj točci dalje se definira svaki koordinatni sustav alata. Upravljačka jedinica ima mogućnost definiranja do deset koordinatnih sustava alata. Koordinatni sustav alata definira se pomakom po tri osi i rotacijama od točke prihvata alata.



Slika 11. a) Točka prihvata alata b) Koordinatni sustav središta alata

4. Flash Builder 4.6

Adobe Flash Builder je računalni program s integriranim razvojnim okruženjem, izgrađen na Eclipse platformi napisanoj u Java programskom jeziku. U njemu se mogu jednostavno i brzo izraditi bogate internetske, desktop i mobilne aplikacije. Adobe Flash Builder nudi ugrađene editore koda za MXML i ActionScript programske jezike. Također nudi interaktivni program za ispravljanje pogrešaka, omogućujući programerima da prođu kroz izvršavanje koda korak po korak, provjeravajući varijable i izraze. Flash Builder Education verzija programa je dostupna besplatno za nekomercijalnu uporabu studentima i nezaposlenim programerima.

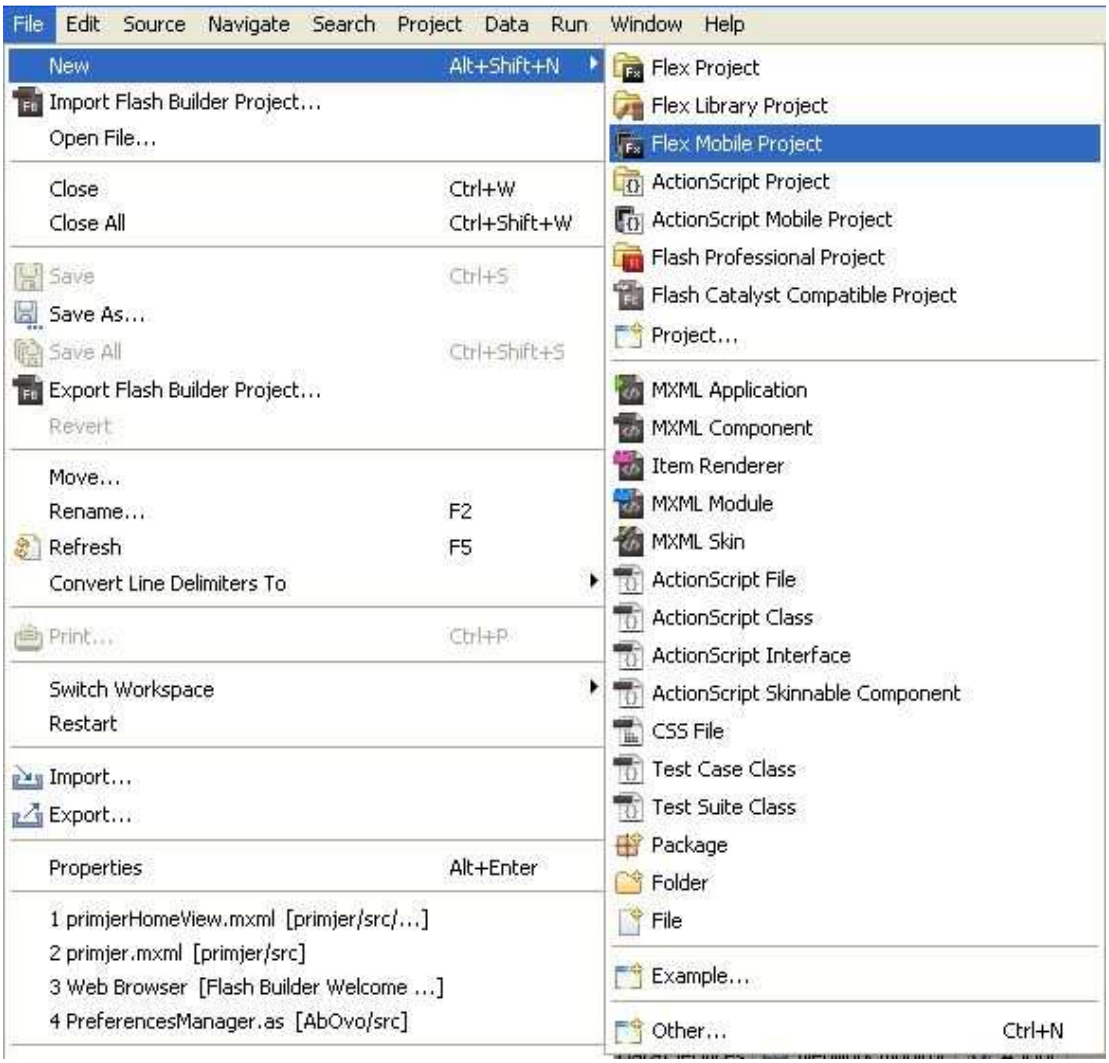


Slika 12. Flash Builder 4.6 - razvojno okruženje

Mobilna aplikacija za upravljanje robotom pomoću iPad uređaja izrađena je u Flash Builder 4.6 programu. Najbolji dio Flash Builder 4.6 programa je potpuna integracija mobilnih razvojnih alata za iOS, Android i BlackBerry operacijske sustave.

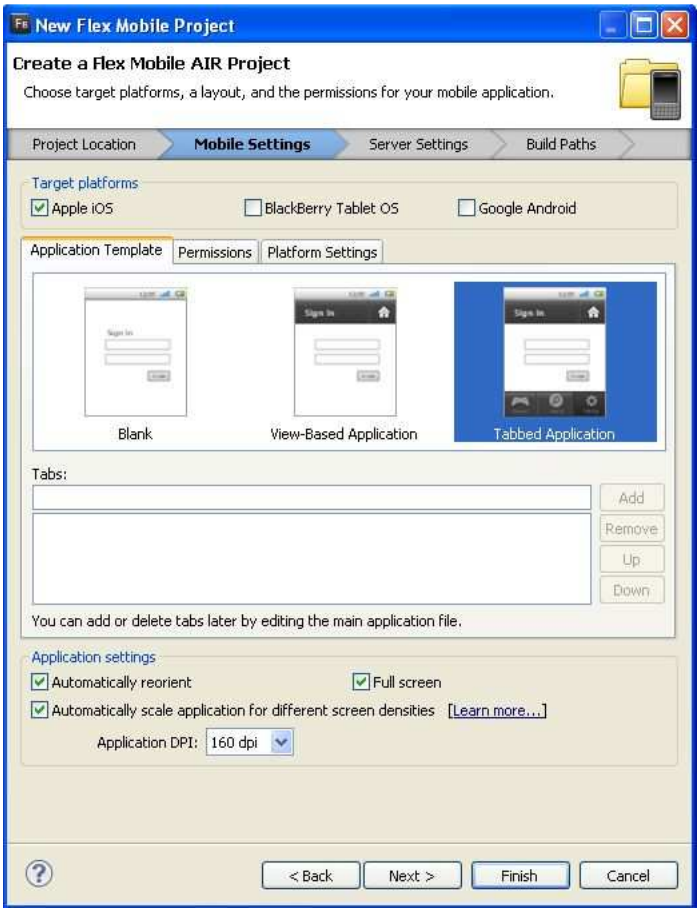
4.1. Kreiranje Flex Mobile projekta

Nakon instalacije i pokretanja programa potrebno je kreirati novi projekt te definirati njegove postavke.

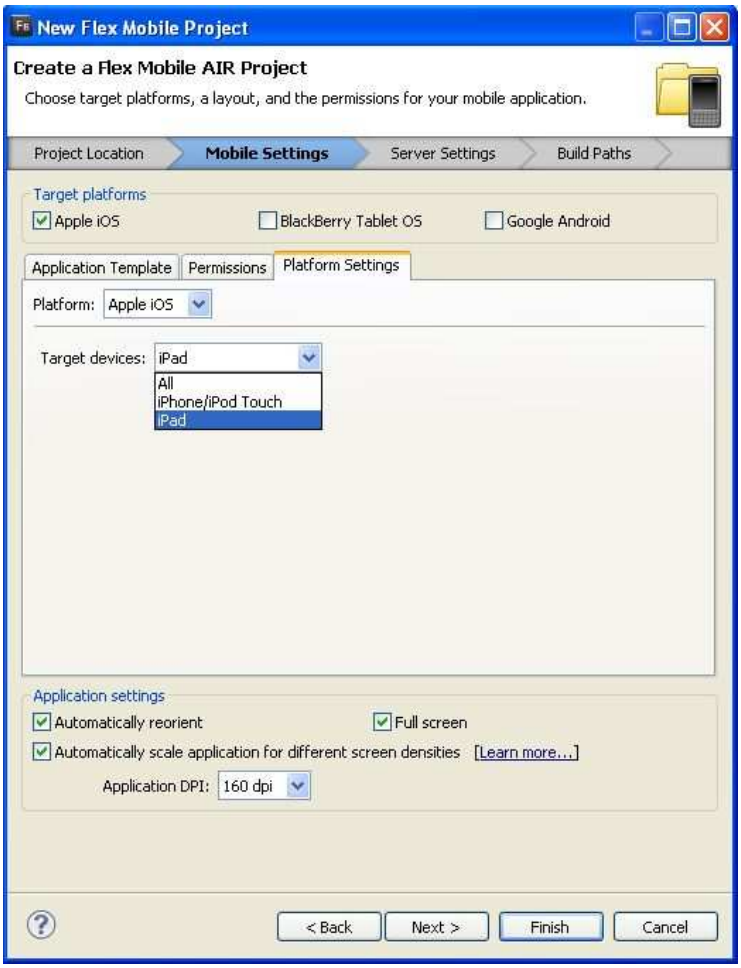


Slika 13. Kreiranje novog Flex Mobile projekta

Slika 14. Definiranje naziva projekta

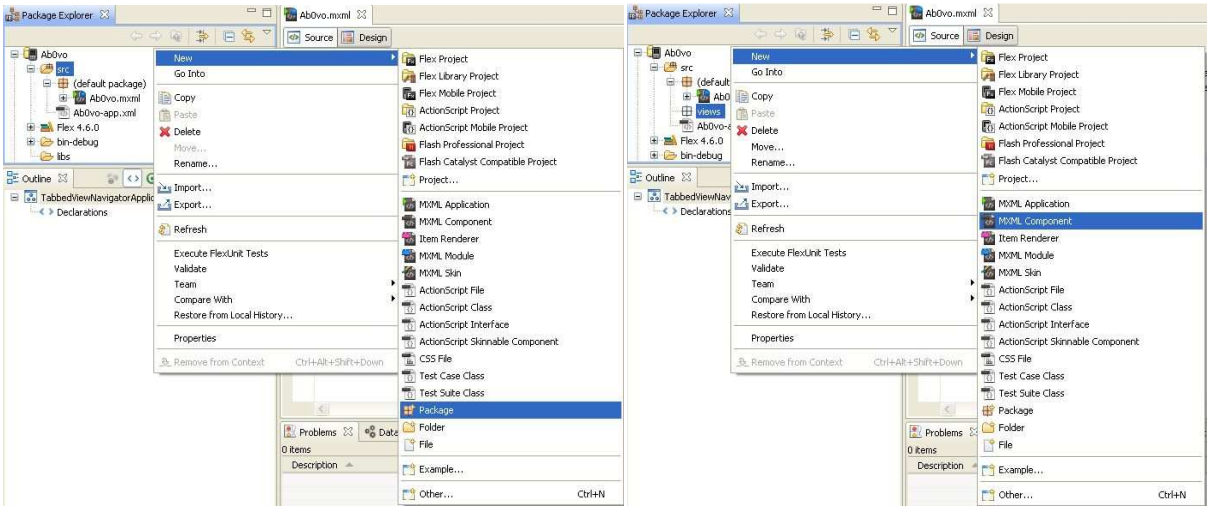


Slika 15. Definiranje ciljanih platformi i izgleda aplikacije



Slika 16. Definiranje uređaja →(*Finish*)

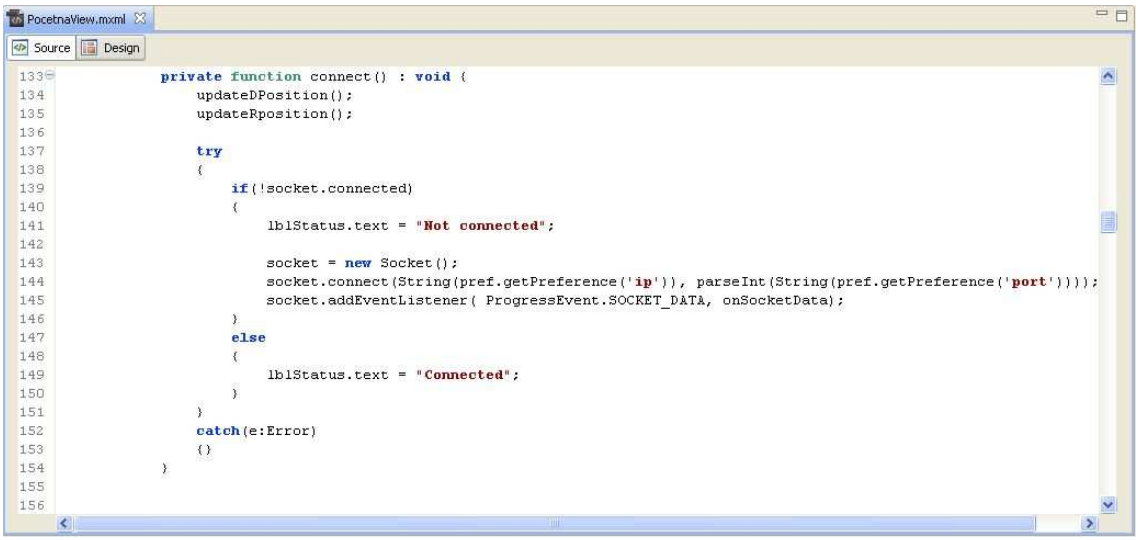
S obzirom na funkciju aplikacije potrebno ju je tako i dizajnirati. AbOvo aplikaciji definirana su tri pogleda što je postignuto kreiranjem tri MXML komponente koje su međusobno povezane programskim kodom.



Slika 17. Kreiranje MXML komponenti

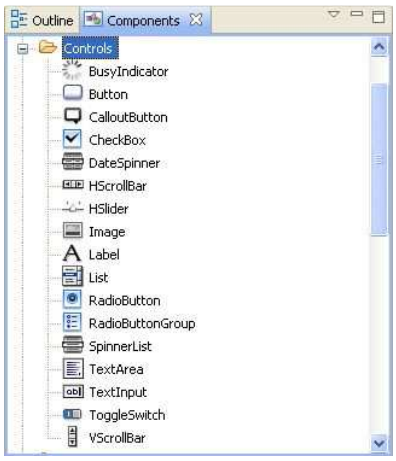
4.1.1. PocetnaView.mxml

PocetnaView.mxml komponetna zamišljena je kao glavni izbornik aplikacije. Pisanjem programskog koda u nju su postavljene sve funkcijske tipke potrebne za upravljanje robotom.



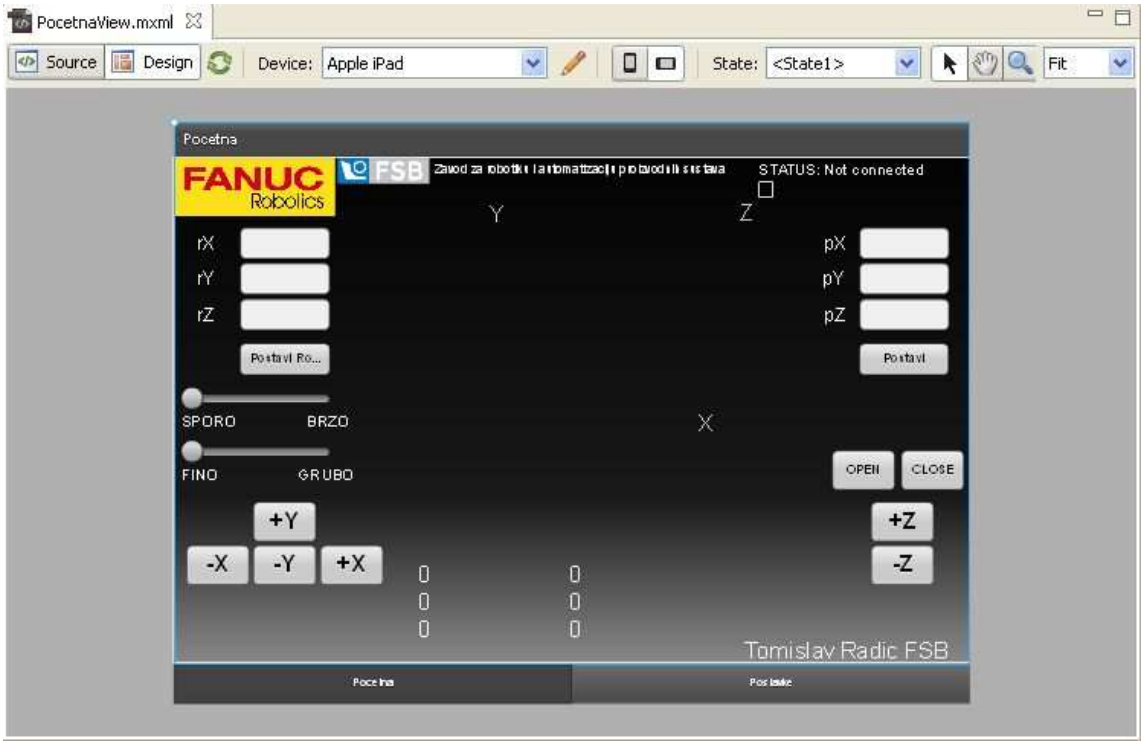
Slika 18. Razvojno okruženje programskog koda

Postoji mogućnost prebacivanja razvojnog okruženja iz izvornog (*Source*) u *Design* mod. *Design* mod nam omogućuje jednostavnije i interaktivnije oblikovanje aplikacije. U njemu odmah vidimo kako aplikacija izgleda te gdje se nalaze kontrole ili tekst. Kontrole jednostavno povućemo mišem iz *Components* kontejnera i postavimo gdje želimo. Programski kod se automatski generira za svaku kontrolu.



Slika 19. Kontrole dostupne u *Design* modu

U izradi ove aplikacije korišrene su sljedeće kontrole: *Button, Hslider, Label, Image* i *TextInput*.



Slika 20. PocetnaView.mxml u *Design* modu

Cijeli programski kod PocetnaView.mxml komponente nalazi se u prilogu završnog rada.

4.1.2. PostavkeView.mxml

Ova komponenta programskog koda ćini pogled u kojem su dostupne postavke aplikacije za povezivanje s upravljanim robotom.

Cijeli programski kod PostavkeView.mxml komponente nalazi se u prilogu završnog rada.

4.1.3.SpecView.mxml

SpecView.mxml komponenta sadrži samo osnovne tehničke karakteristike upravljanog robota i nema drugu svrhu.

Programski kod SpecView.mxml komponente nalazi se u prilogu završnog rada.

4.1.4. PreferencesManager.as

PreferencesManager.as je ActionScript datoteka koja sadrži programski kod koji upravlja postavkama iz SpecView.mxml komponente

Programski kod PreferencesManager.as datoteke nalazi se u prilogu završnog rada.

4.1.5. RobotVisualization.as

ActionScript datoteka s porgramskim kodom koji iz povratne veze upravljačke jedinice robota na zaslonu iPad uređaja iscrtava položaj robotske ruke u prostoru.

Programski kod RobotVisualization.as datoteke nalazi se u prilogu završnog rada.

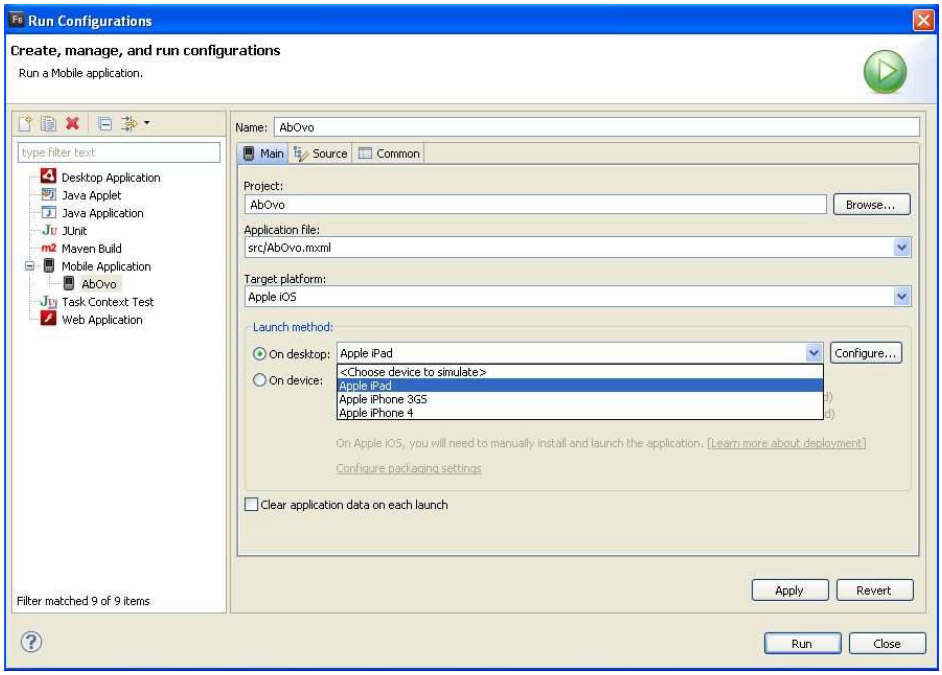
Sve MXML komponente i ActionScript datoteke međusobno izmjenjuju informacije te kao cjelina tvore AbOvo aplikaciju.

4.2. Simulacija iPad uređaja

Flash Builder 4.6 u mogućnosti je simulirati uređaje za koje se izrađuju aplikacije te tako testirati još ne dovršenu aplikaciju. Za testiranje izgleda i funkcionalnosti AbOvo aplikacije simuliramo iPad uređaj. Pritiskom na tipku *Run Configurations* dobivamo izbornik za odabir projekta koji želimo pokrenuti.

Slika 21. *Run*

Izabiremo metodu pokretanja *On desktop* i uređaj za simulaciju te pokrećemo konfiguraciju.



Slika 22. *Run Configurations* izbornik

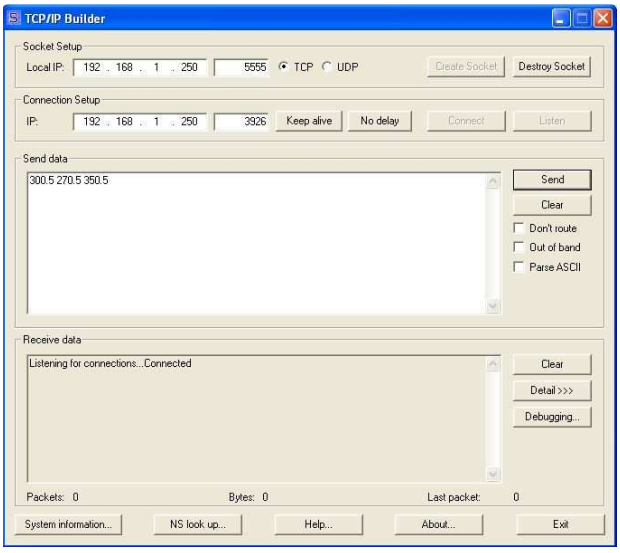
Tako pokrenuta aplikacija radi isto kao i na stvarnom uređaju. Ako smo zadovoljni izgledom aplikacije preostaje nam još testirati funkcionalnost tipki. Zamišljeno je da pritiskom na neku od tipki aplikacija šalje naredbu u obliku stringa putem TCP/IP protokola na upravljačku jedinicu robota. Upravljčka jedinica nakon što primi naredbu vrši neku operaciju robotom.

4.2.1. TCP/IP protokol

TCP/IP je skup protokola koji se koriste za povezivanje računala i srodnih uređaja putem *socketa*. Za *socket* možemo reći da je pristupna točka definirana IP adresom uređaja i odabranim portom. Služi za komunikaciju između aplikacija na različitim računalima.

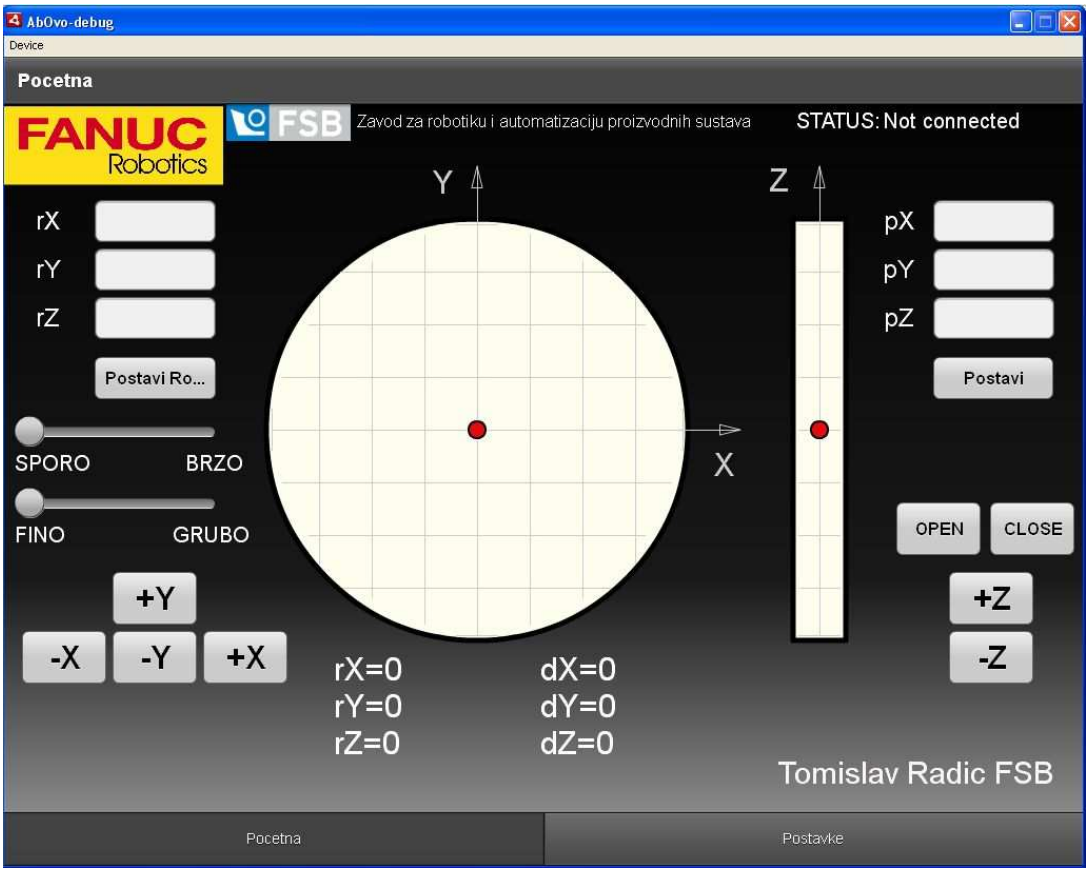
4.2.2.TCPIP Builder

Da bi provjerili da li se naredba uopće šalje možemo se poslužiti TCPIP Builder programom. TCPIP Builder je je besplatni program za testiranje *socketa*. Može djelovati kao klijent ili server. Postavimo li ga kao server u lokalnoj mreži našeg računala na proizvoljnom portu i pokrenemo simulaciju aplikacije u Flash Builder-u doći će do povezivanja s aplikacijom kao klijentom.



Slika 23. TCPIP Builder

Pritiskom na neku komandu u simuliranoj AbOvo aplikaciji naredba se treba pojaviti u prozoru *Receive data*, TCPIP Builder programa. Upisivanjem stringa definiranog u aplikaciji, u prozor *Send data* TCPIP Builder programa i pritiskom na tipku *Send* testiramo povratnu vezu AbOvo aplikacije. Ako su primljeni podatci pravilno interpretirani uspješno smo testirali aplikaciju.

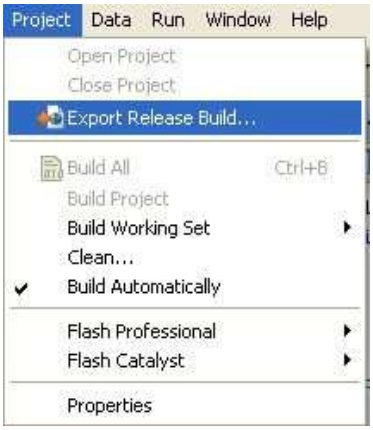


Slika 24. Simulacija AbOvo aplikacije

4.3. Sastavljanje gotove aplikacije i prebacivanje na iPad uređaj

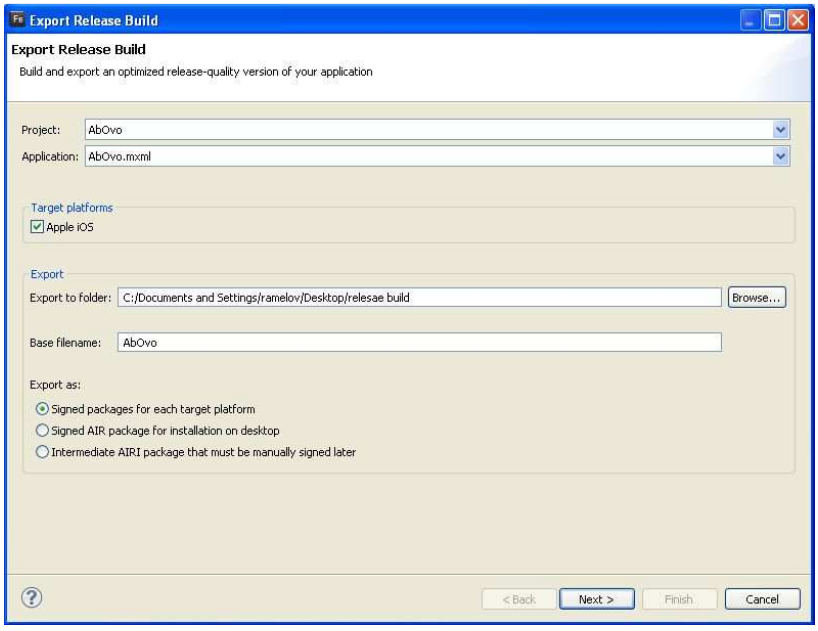
Nakon uspješnog testiranja svih komandi i povratne veze AbOvo aplikaciju je potebno spakirati u jednu datoteku iOS App oblika. Takva datoteka može se prebaciti na iPad uređaj preko iTunes programa.

U *Project* padajućem izborniku Flash Builder programa izaberemo Export Release Build naredbu.



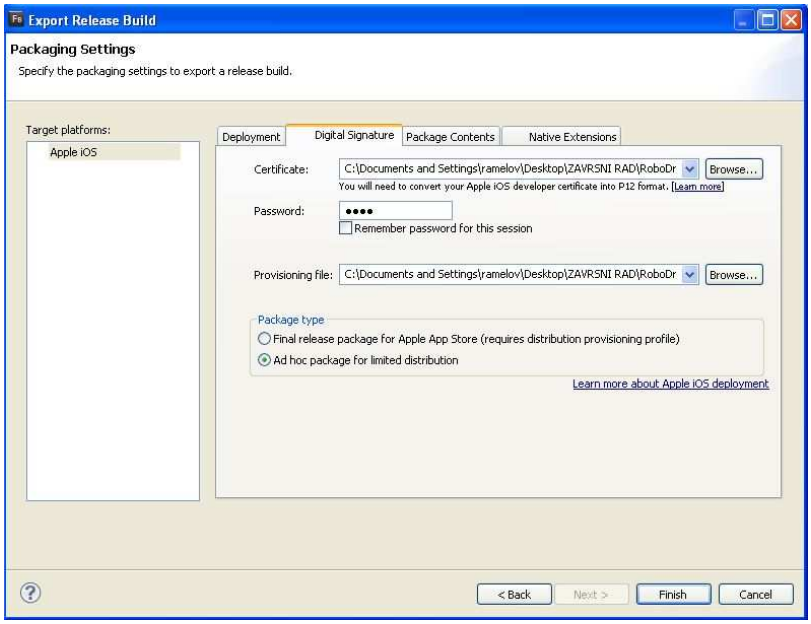
Slika 25. *Project* izbornik

Zatim izabiremo projekt koji želimo spakirati u iOS App datoteku te mjesto gdje će ona biti spremljena.



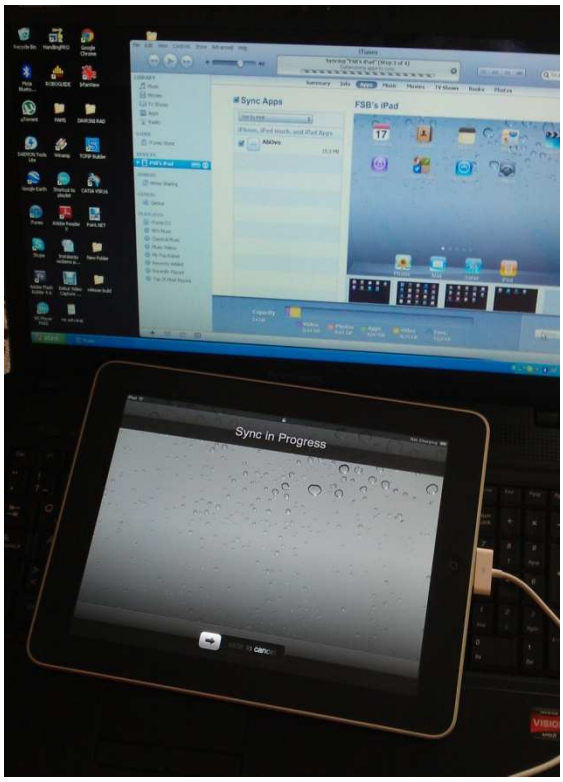
Slika 26. *Export Release Build* izbornik

Da bi završili ovaj postupak sastavljanja i pakiranja aplikacije potreban joj je digitalni potpis. Za stavljanje digitalnog potpisa potrebno je imati dvije datoteke. Jedna sadrži certifikat (*Cetificate*) a druga odobrenje (*Provisioning file*). Certifikat i odobrenja možete dobiti ako Apple-u platite 99$ (američkih dolara) te tako postanete njihov razvojni programer.



Slika 27. Uvjeti za digitalni potpis

Nakon spajanja iPad uređaja sa računalom i sinkronizacije sa iTunes programom, gotovu iOS App datoteku jednostavno povućemo mišem u prostor iTunes-a. Tjekom ponovne sinkronizacije sve aplikacije koje se nalaze u iTunes programu se instaliraju na iPad uređaj te je moguće njihovo korištenje.



Slika 28. Instalacija aplikacije na iPad uređaj

Slika 29. Instalirana aplikacija

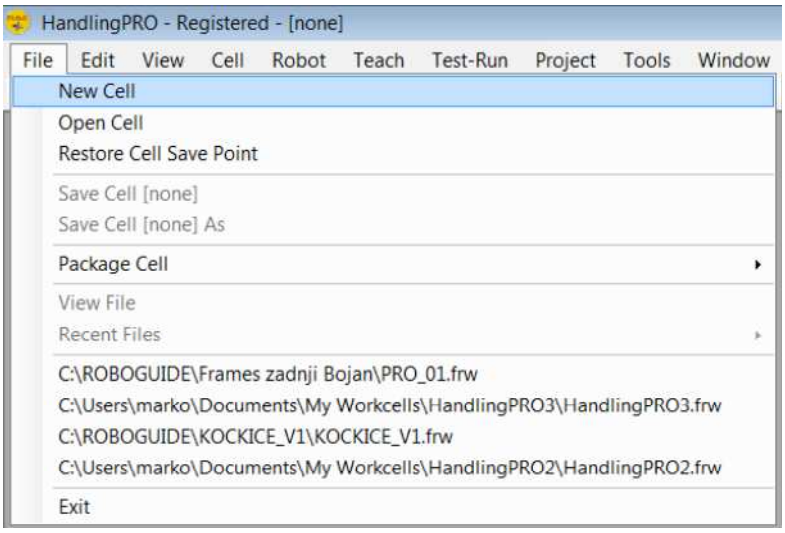
5. Roboguide

Roboguide je simulacijski paket koji služi za *offline* programiranje Fanuc robota. Donosi dodatne funkcionalnosti u odnosu na rad sa upravljačkom konzolom. Moguća je izrada i testiranje programa na osobnom računalu bez potrebe za zaustavljanjem rada robota. Pri školovanju većeg broja ljudi nema potrebe za velikim brojem realnih robota. Unutar Roboguide-a nalazi se virtualni sustav koji sadrži virtualne mehaničke jedinice robota i njihove upravljačke jedinice te upravljačke konzole.

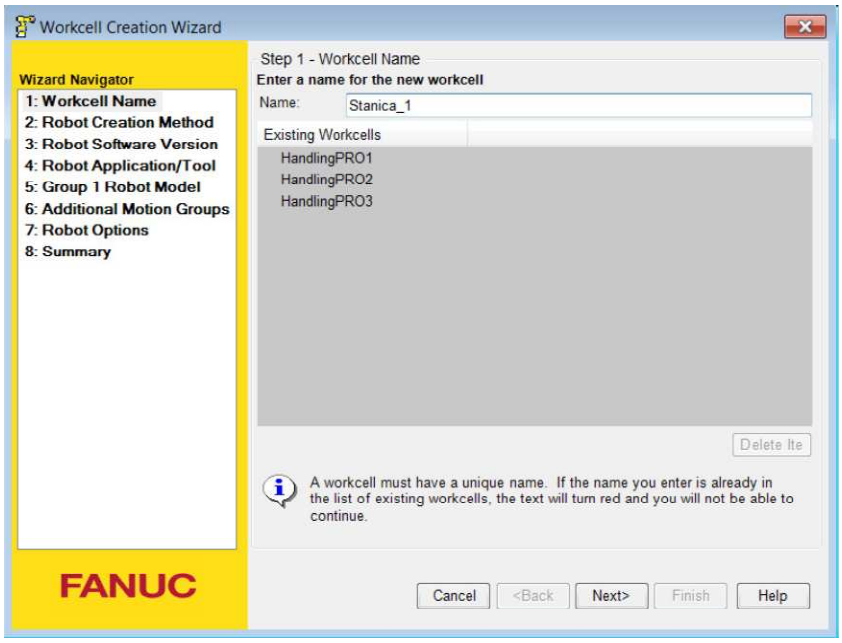
Roboguide u sebi sadrži i proceduralni programski jezik KAREL u kojem je izrađen program za ovaj završni rad. Program može interpretirati naredbe zadane od strane AbOvo aplikacije te slati podatke o poziciji robotske ruke u realnom vremenu.

5.1. Izrada robotske stanice u Roboguide-u

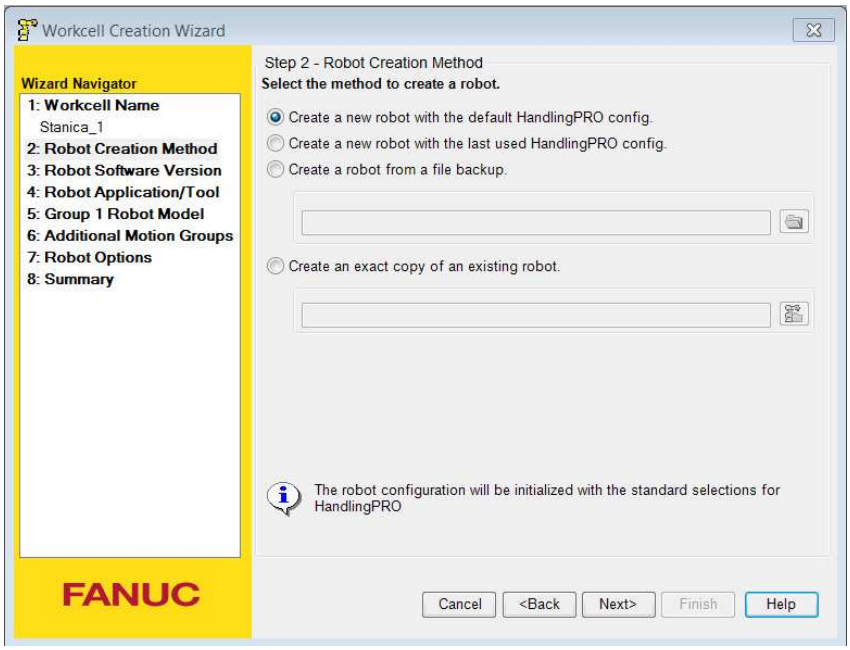
Prije samog rada u Roboguide-u potrebno je definirati postavke robotske stanice te izvršiti izbor potrebnih parametara.



Slika 30. Kreiranje nove robotske stanice

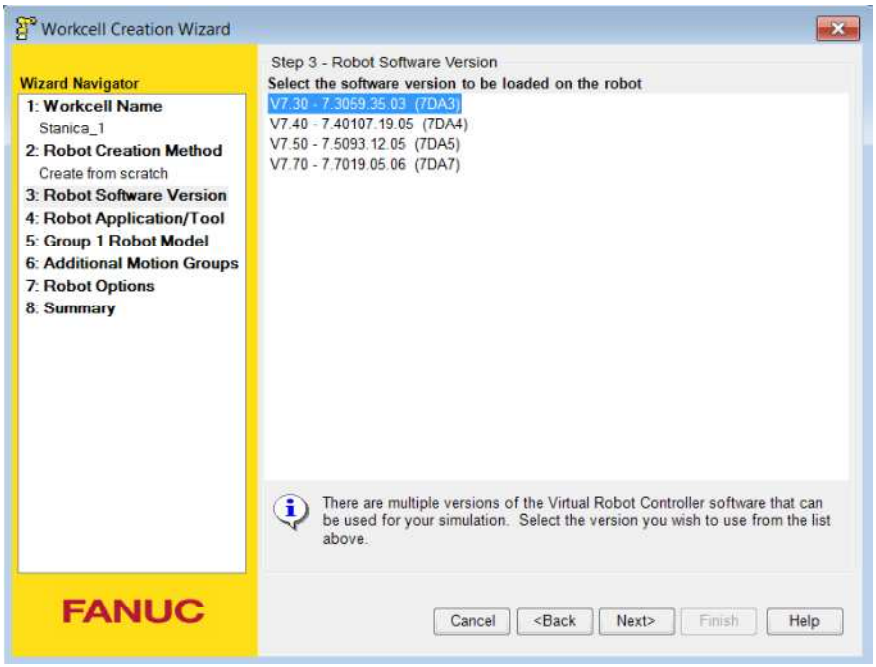


Slika 31. Definiranje naziva robotske stanice

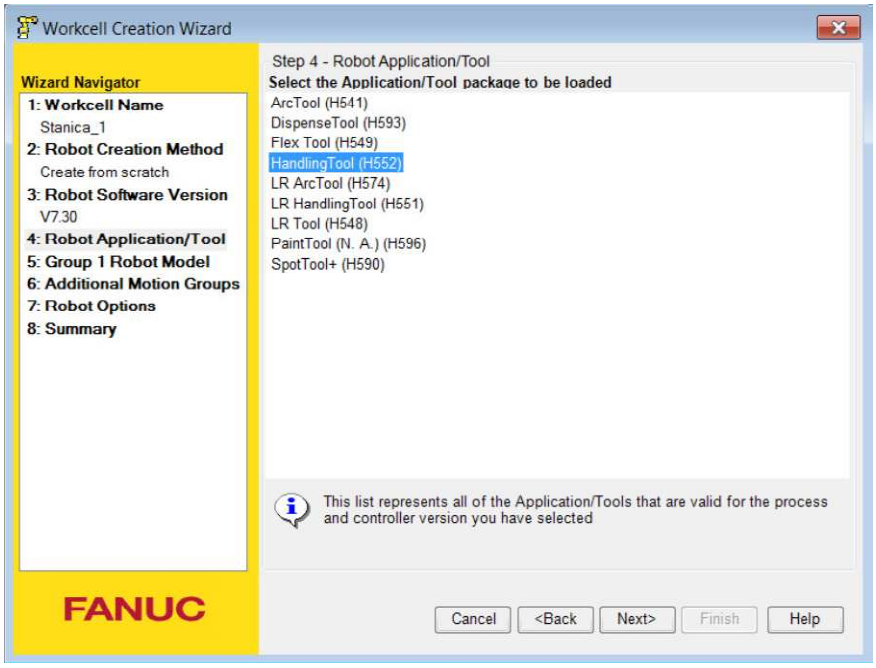


Slika 32. Definiranje metode

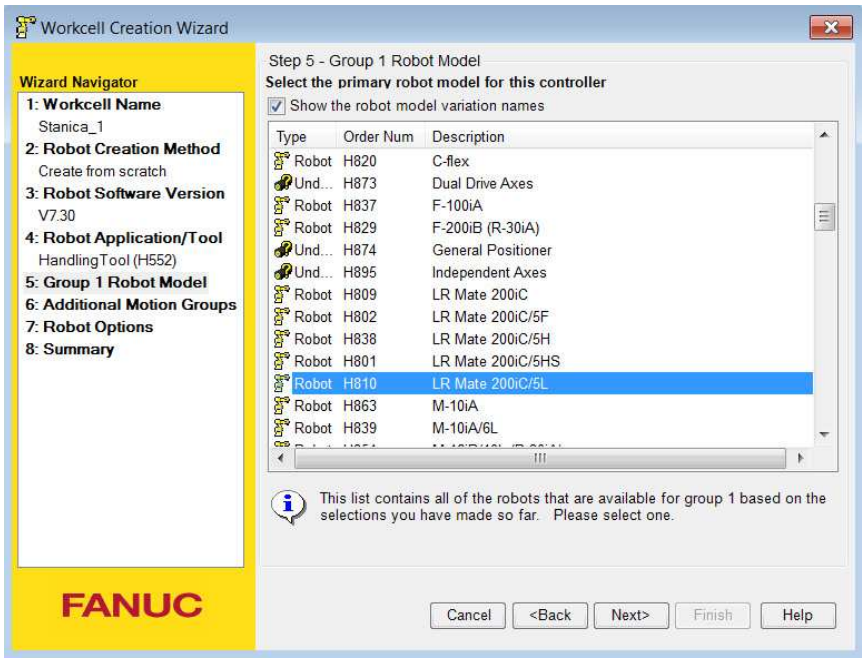
Prilikom izbora verzije softvera upravljačke jedinice robota poželjno je izabrati verziju 7.30 iz razloga što će KAREL programi pisani za tu verziju softvera biti kompatibilni sa novijim verzijama dok suprotno nije moguće.



Slika 33. Izbor verzije softvera upravljačke jedinice robota

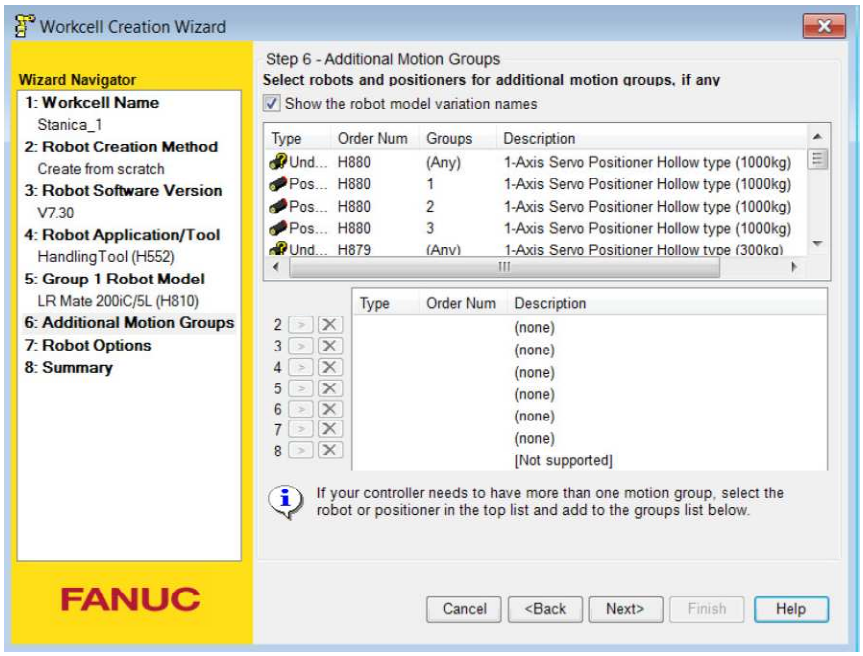


Slika 34. Izbor aplikacije – *Handling Tool*



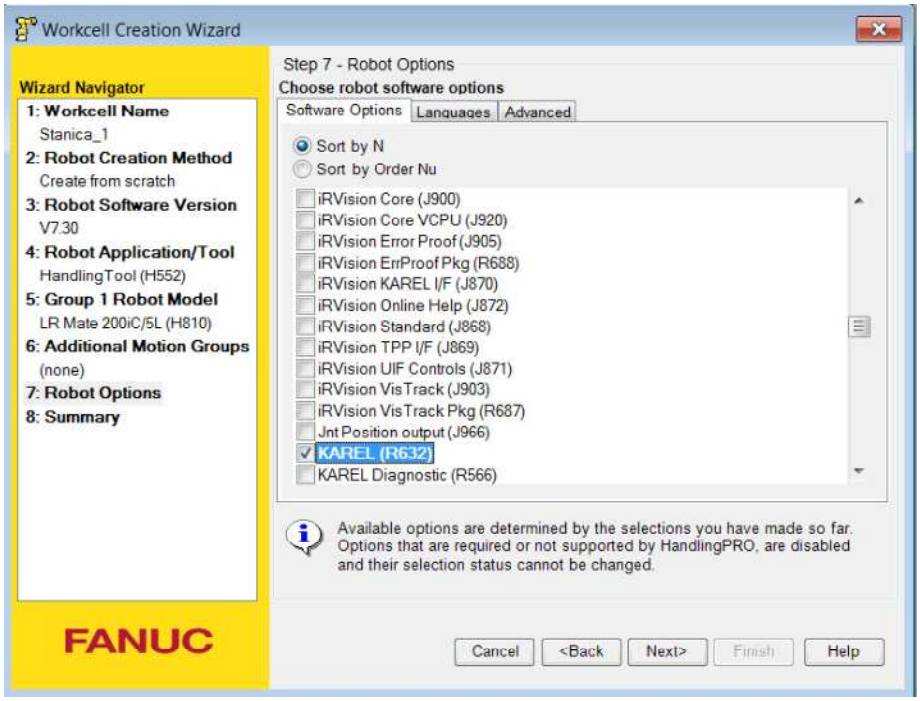
Slika 35. Izbor mehaničke jedinice robota

Prilikom definiranja dodatnih *motion* grupa nije ih potrebno specificirati. Ova se opcija koristi ako se želi na jednu upravljačku jedinicu robota povezati dodatne mehaničke jedinice. Tada se one pojavljuju kao dodatne *motion grups* u upravljačkoj jedinici robota.

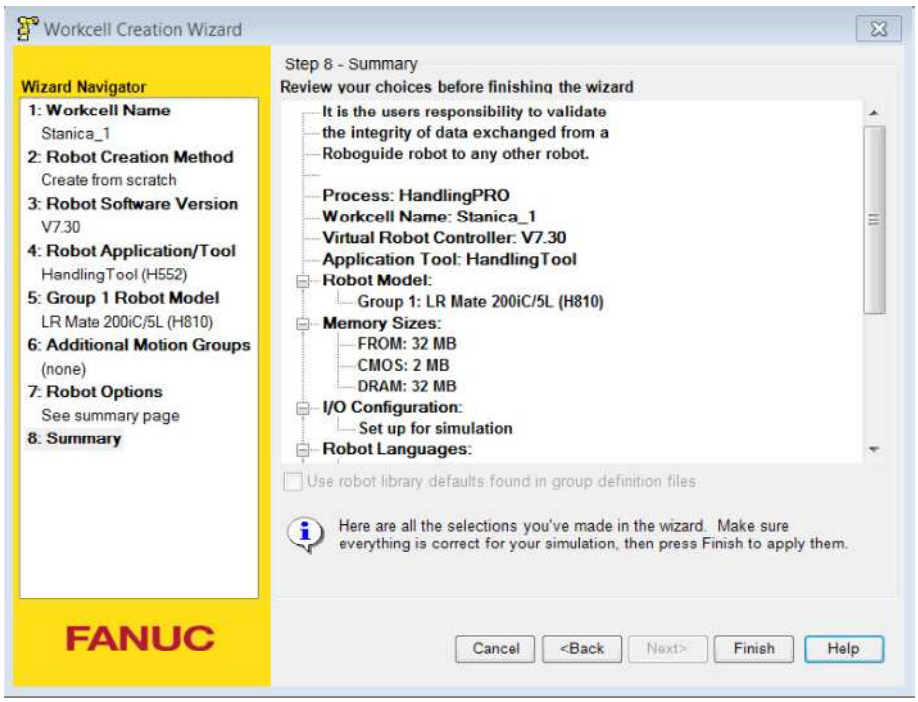


Slika 36. Definiranje dodatnih grupa

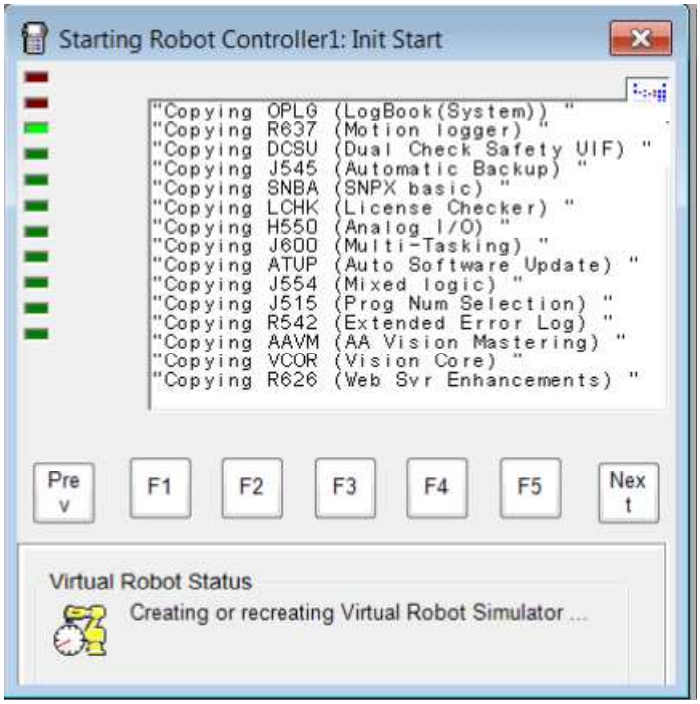
Prilikom definiranja softverske konfiguracije upravljačke jedinice robota potrebno je izabrati opciju KAREL (R632) kako bi se omogućilo programiranje pomoću programskog jezika KAREL.



Slika 37. Definiranje softverske kofiguracije upravljačke jedinice robota

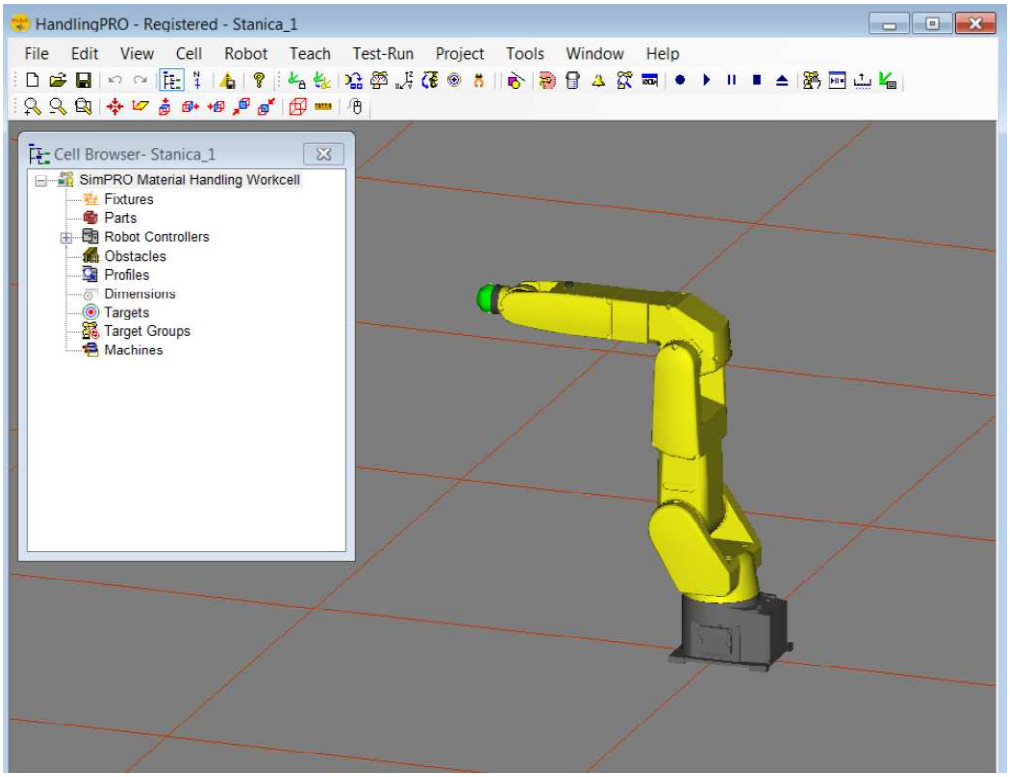


Slika 38. Sažetak konfiguracije robotske stanice



Slika 39. Inicijalizacija virtualne upravljačke jedinice

Nakon inicijalizacije upravljačke jedinice te određenih poruka na ekranu se pojavljuje radno sučelje Roboguide-a.



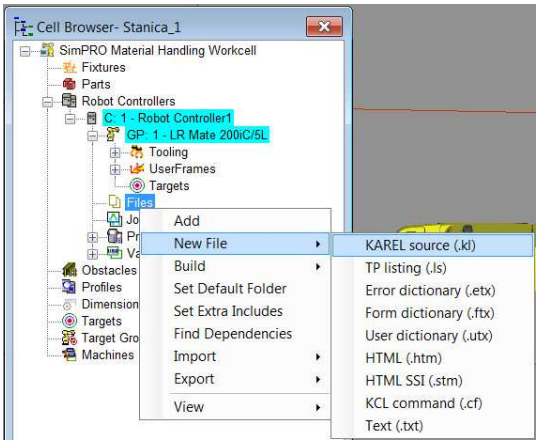
Slika 40. Sučelje programskog paketa Roboguide

5.2. Kreiranje KAREL programa

Karel je proceduralni jezik koji služi za programiranje Fanuc robotskih sruktura. Kod izvršavanja programa unutar simulacijskog paketa Roboguide postoje određena ograničenja u odnosu na realnu upravljačku jedinicu. Naime nije moguće izvršavanje programa koji koriste komuniciranje pomoću *socketa – Socket messaging*. Testiranje programa potrebno je izvršiti na realnom robotu. Za testiranje korišten je FanucLRM200iC/5L u Labaratoriju za projektiranje izradbenih i montažnih sustava.

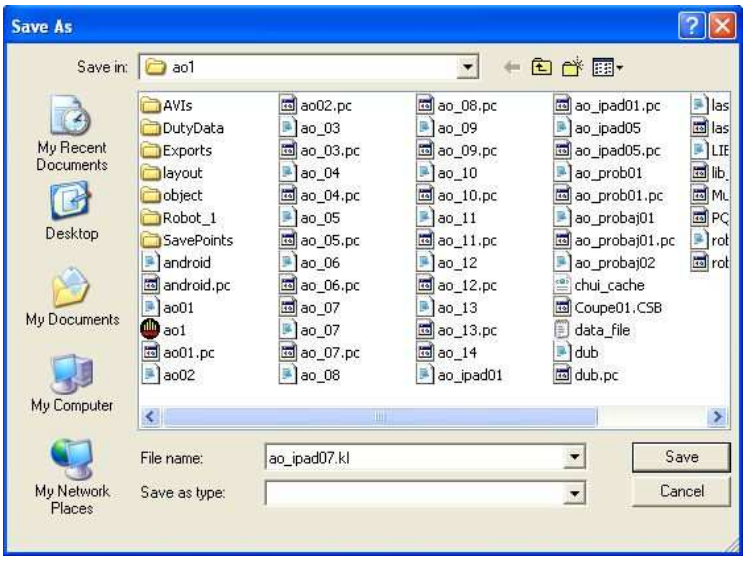
5.2.1. Pisanje programa

Za pisanje Karel programa potrebno je iz preglednika robotske stanice kreirati novu KAREL datoteku



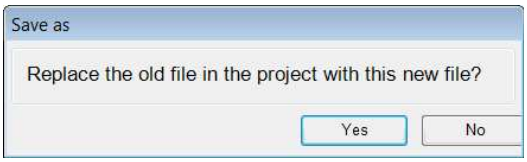
Slika 41. Dodavanje nove KAREL datoteke

Otvaranjem editora potrebno je sačuvati Karel datoteku. Datoteku je potrebno imenovati sa ekstenzijom kl.



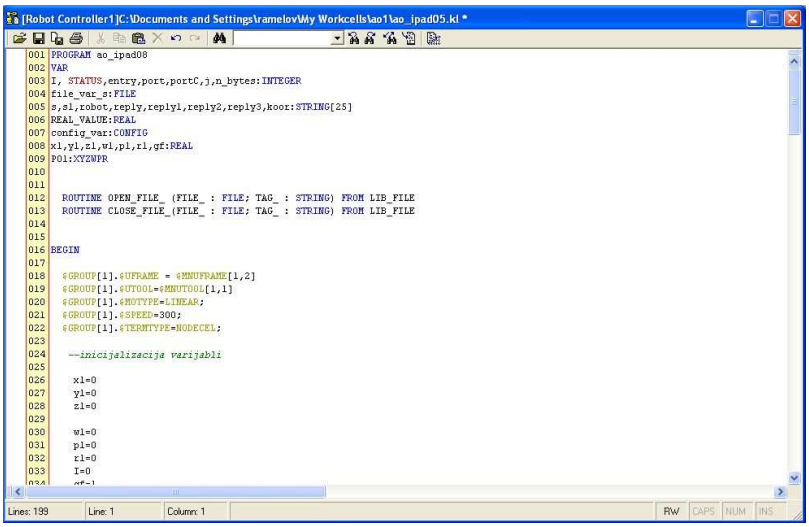
Slika 42. Spremanje novog KAREL programa

Na sljedeći upit potrebno je odgovoriti sa *Yes*.



Slika 43. Upit

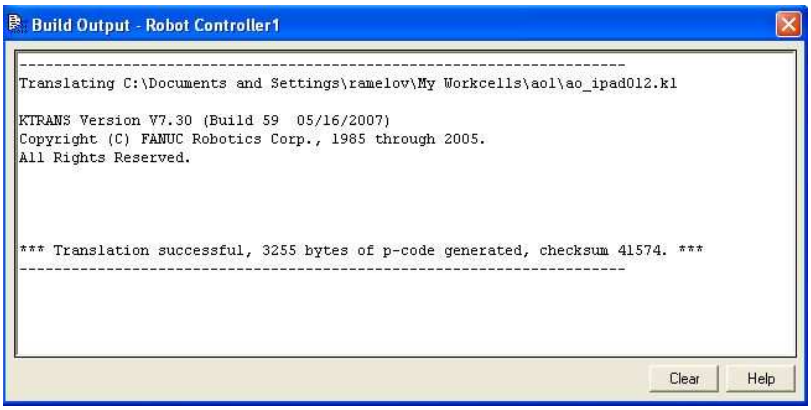
Nakon uspješnog kreiranja Karel datoteke imamo otvoren editor u koji pišemo programski kod. Sintaksa programskog jezika vrlo je slična programskom jeziku Pascal.



Slika 44. Sučelje za pisanje KAREL programa

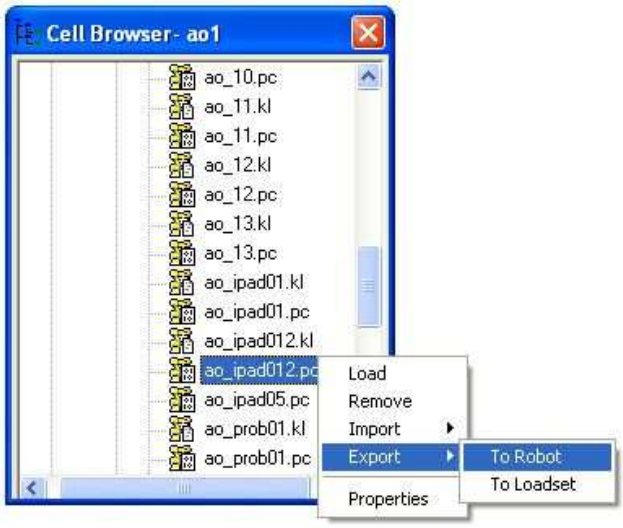
Cijeli programski kod KAREL programa nalazi se u prilogu završnog rada.

Napisani program potrebno je prevesti u datoteku koju šaljemo na upravljačku jedinicu robota. Pritiskom na tipku *Build* pojavljuje se sljedeća obavjest ukoliko nije bilo grešaka u sintaksi.



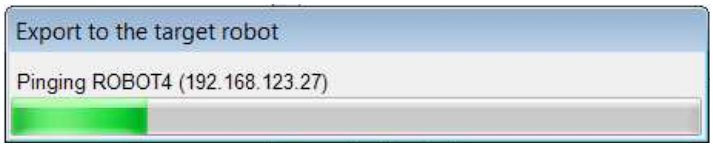
Slika 45. Prevođenje programa

Sada se u pregledniku robotske stanice nalazi nova datoteka sa ekstenzijom .pc. To je datoteka koju je potrebno eksportirati na upravljačku jedinicu robota.



Slika 46- Kopiranje prevedenog programa na upravljačku jedinicu realnog robota - izbor

Izborom željenog robota u robotskom susjedstvu (*Robot Neighborhood*) te potvrdom na tipku *Export* označena pc datoteka kopira se na upravljačku jedinicu realnog robota.



Slika 47. Kopiranje prevedenog programa na upravljačku jedinicu realnog robota – slanje

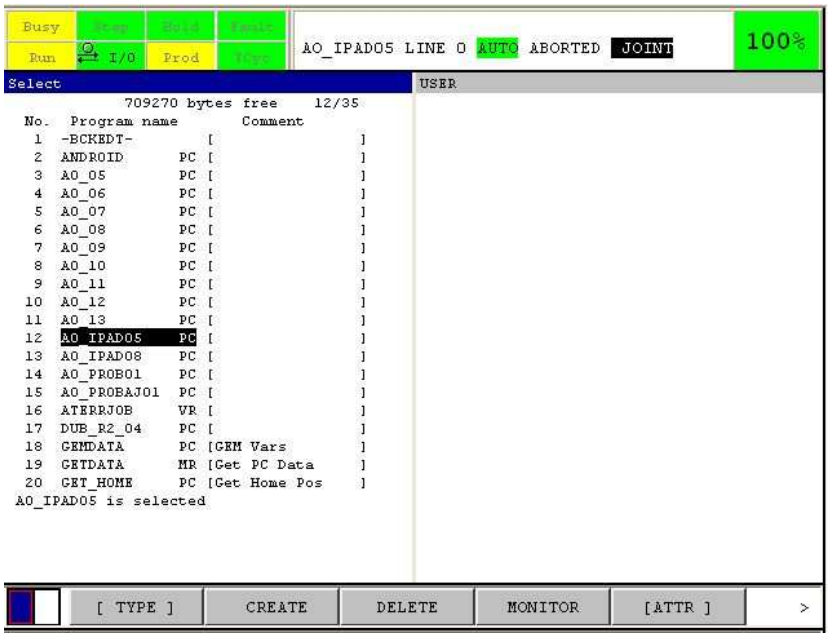
5.2.2. Pokretanje programa

Za uspješno pokretanje i izvođenje programa na upravljačkoj konzoli robota sljedeći uvjeti moraju biti zadovoljeni:

1. Upravljačka konzola upaljena – ON
2. *Deadman switch (DS)* – tri položaja – samo srednji položaj omogućava ručni rad

3. Prekid postojećeg programa – izvršava se tipkom Fctn te izborom
(1) Abort all
4. Poništenje grešaka – kombinacija tipki SHIFT + Reset
5. Program mora biti označen pritiskom na tipku *Select*
upravljačke konzole
6. Gljiva na upravljačkoj konzoli i upravljačkoj jedinici izvučena
7. Postavka rada u T1 (max 250mm/s)!

Kombinacijom tipki SHIFT i DISP moguće je ekran upravljačke konzole podjeliti na 1,2 ili 3 podekrana koji služe za povećanu preglednost



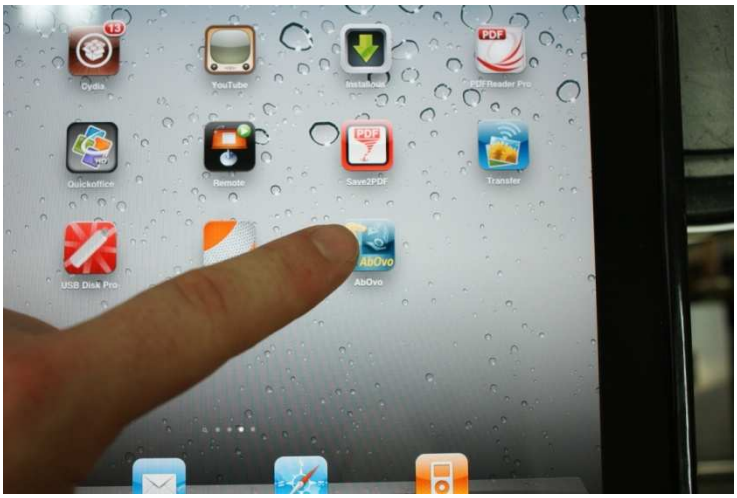
Slika 48. Upravljačka konzola spremna za pokretanje programa

Program se pokreće kombinacijom tipki SHIFT i FWD sa upravljačke konzole robota. Program AO_IPAD09 napisan kao praktični dio ovog završnog rada učinit će upravljačku jedinicu robota poslužiteljem (serverom). Na IP adresi robota '192.168.123.27' i odabranom portu '10267' otvorit će se *socket* za komunikaciju s iPad uređajem. Nakon uspostavljanja veze moguće je bežićno upravljanje robotom.

6. AbOvo aplikacija

6.1. Izvođenje jednostavne *Pick & Place* operacije

AbOvo aplikaciju možemo pokrenuti dodirom prsta na ikonu aplikacije koja se nalazi u izborniku iPad uređaja.



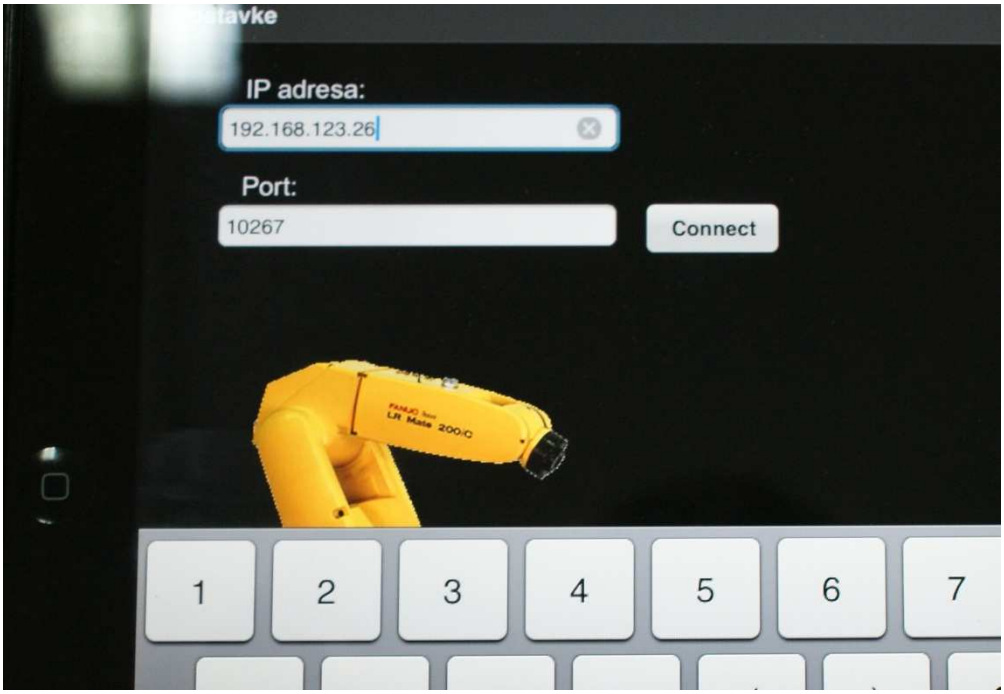
Slika 49. Izbornik iPad uređaja

Prethodno je na upravljačkoj konzoli robota potrebno pokrenuti KAREL program AO_IPAD09 za komunikaciju robota s iPad uređajem.

Slika 50. Upravljačka konzola robota

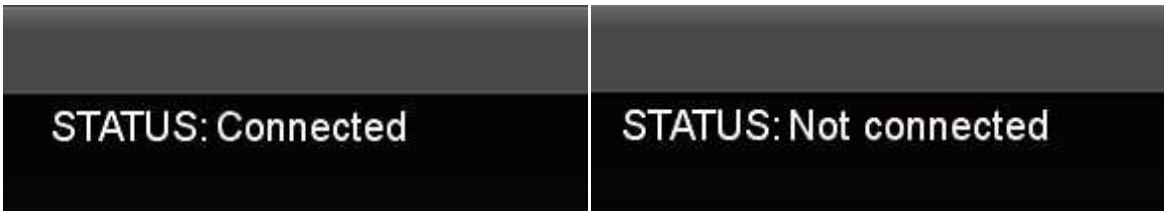
Pokrenuta aplikacija automatski uspostavlja konekciju sa programom na upravljačkoj jedinici robota. Ako se veza ne uspostavi potebno je provjeriti da li je upisana isparavna IP adresa i port upravljačke jedinice robota na koju se spaja aplikacija. IP adresu i port možemo upisati u postavkama aplikacije.

IP adresa robota na kojem je testirana aplikacija glasi: 192.168.123.26. Port na kojem se može uspostaviti *socket* konekcija je 10267. Port je definiran u KAREL programu AO_IPAD09 i njega je moguće promjeniti.



Slika 51. Postavke aplikacije

Status aplikacije nam govori je li veza uspostavljena (Connected) ili ne (Not connected).

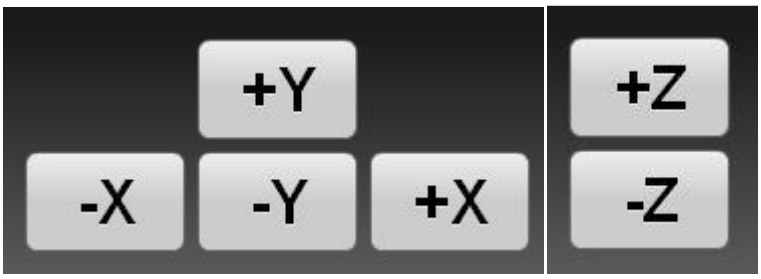


Slika 52. Status aplikacije

Kada je veza uspostavljena možemo upravljati robotom. Upravljanje se svodi na linearno gibanje robota po x, y, z osi koordinatnog sustava alata s obzirom na glavni koordinatni sustav robota. Moguće je zatvarati i otvarati hvataljku, mjenjati brzinu i inkrement gibanja te pozicionirati robota direktnim unosom željenih koordinata u aplikaciju.

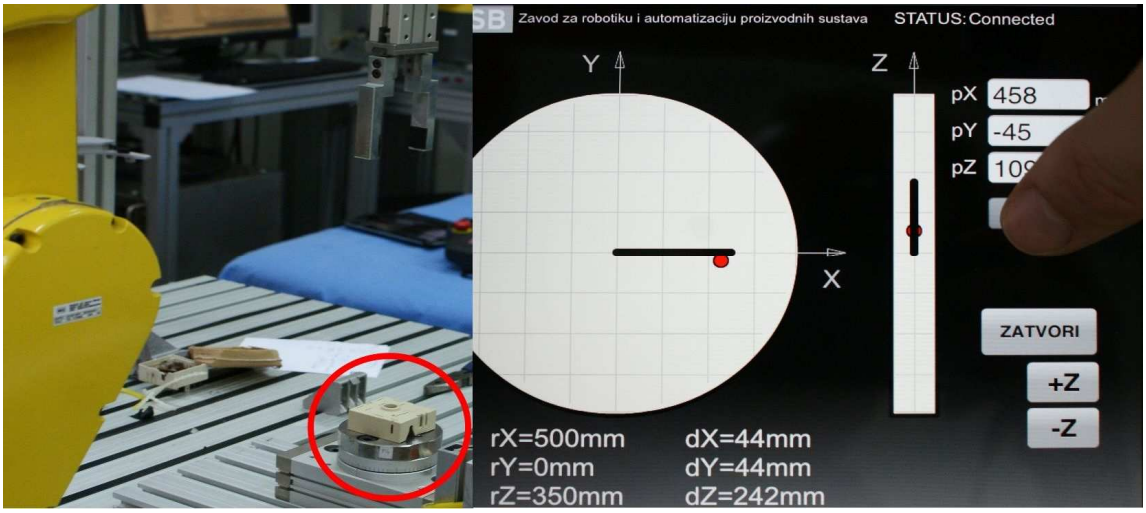
Slika 53. Pokrenuta aplikacija

Za gibanje po koordinatnim osima aplikacija ima šest funkcijskih tipki. Pozitivan i negativan smjer po x, y i z osi.



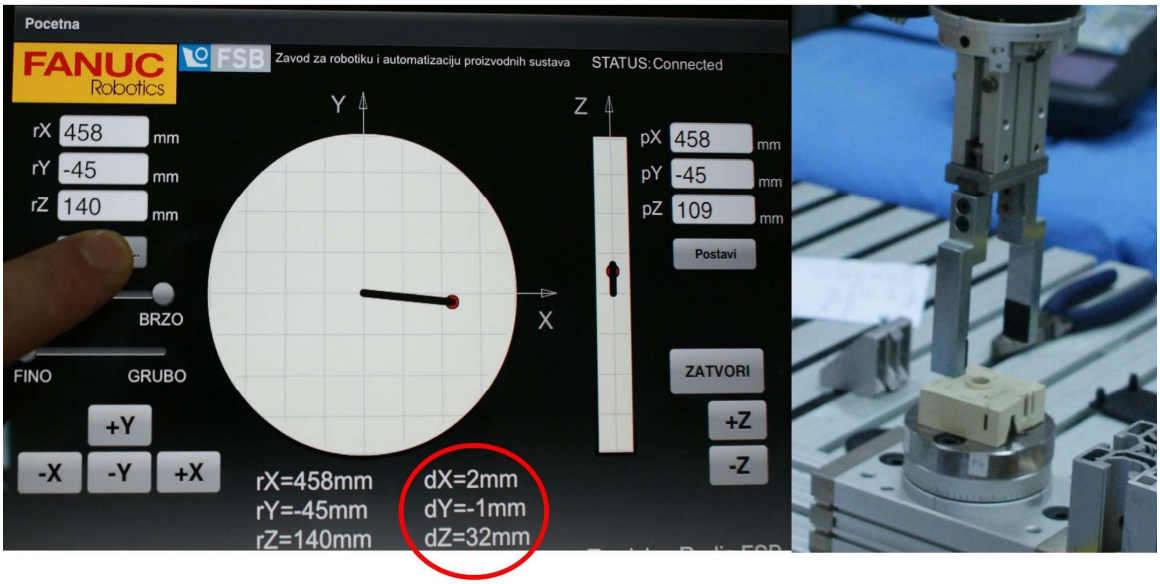
Slika 54. Komande za gibanje robota

Ako su nam poznate točne koordinate nekog predmeta od interesa moguće ih je unijeti u aplikaciju i time virtualno prikazati položaj tog predmeta u radnom prostoru robota.



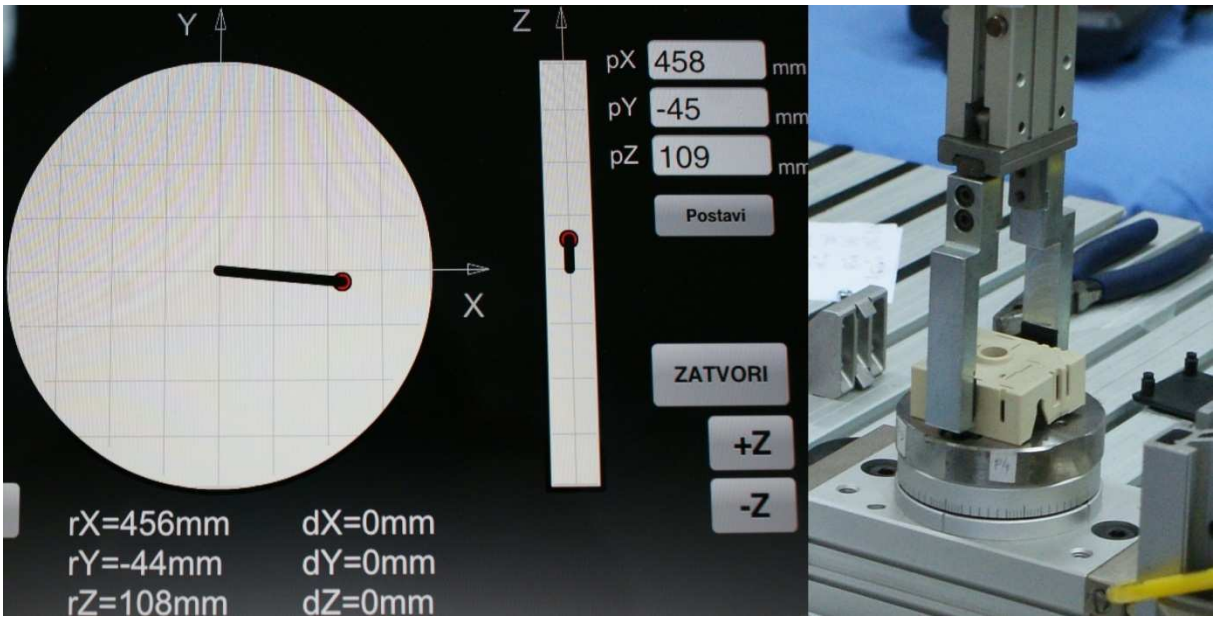
Slika 55. Predmet od interesa

Sljedeći korak je dovođenje hvataljke robota u neposrednu blizinu predmeta od interesa. Za direktno postavljanje robota u željenu poziciju imamo komandu Postavi Robota sa tri prozora za unos koordinata.



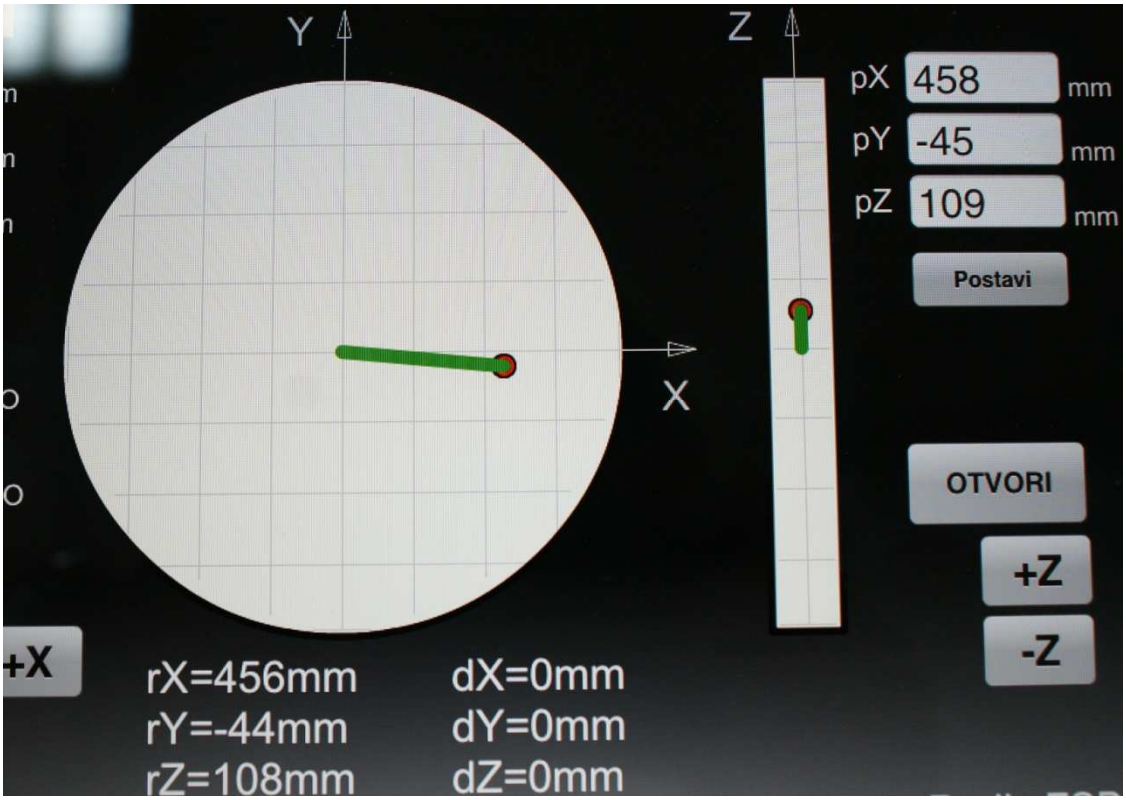
Slika 56. Udaljenosti alata od predmeta od interesa

Na prethodnoj slici (Slika 56.) možemo vidjeti koliko nam je odstupanje (zaokruženo crveno) od potpunog poklapanja hvataljke robota i predmeta od interesa. Za fino pozicioniranje hvataljke koristimo se komanadama za gibanje robota (Slika 53.).



Slika 57. Poklapanje alata i predmeta od interesa

Hvataljku zatvaramo komandom 'ZATVORI'. Nakon pritiska komande njeno stanje mjenja se u 'OTVORI' . Ako smo tom operacijom zahvatili predmet virtualni prikaz robota mjenja boju u zeleno.

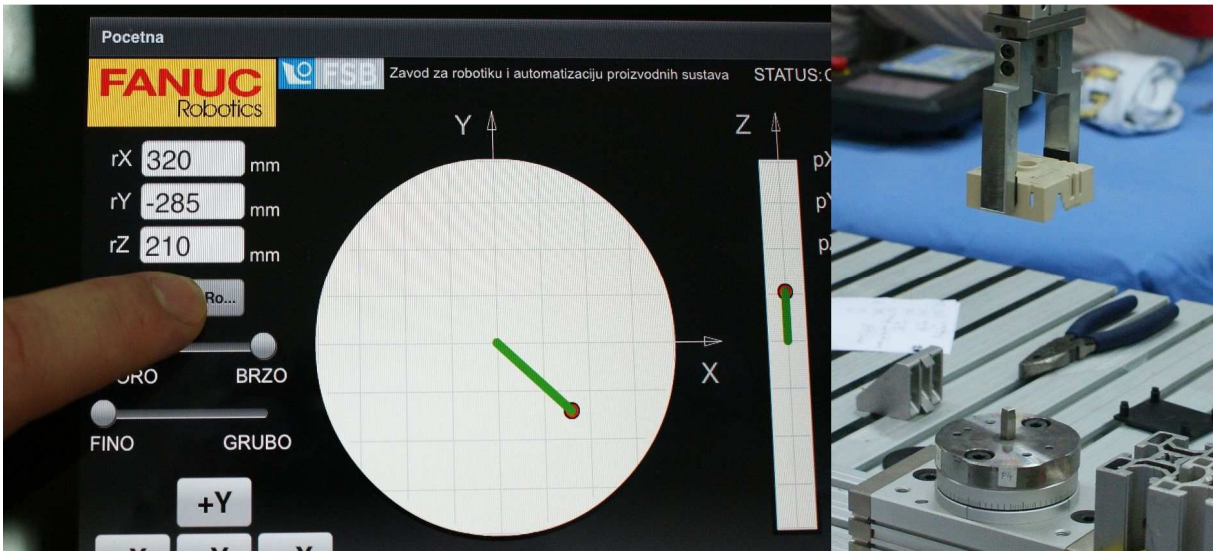


Slika 58. Zahvaćanje predmeta od interesa

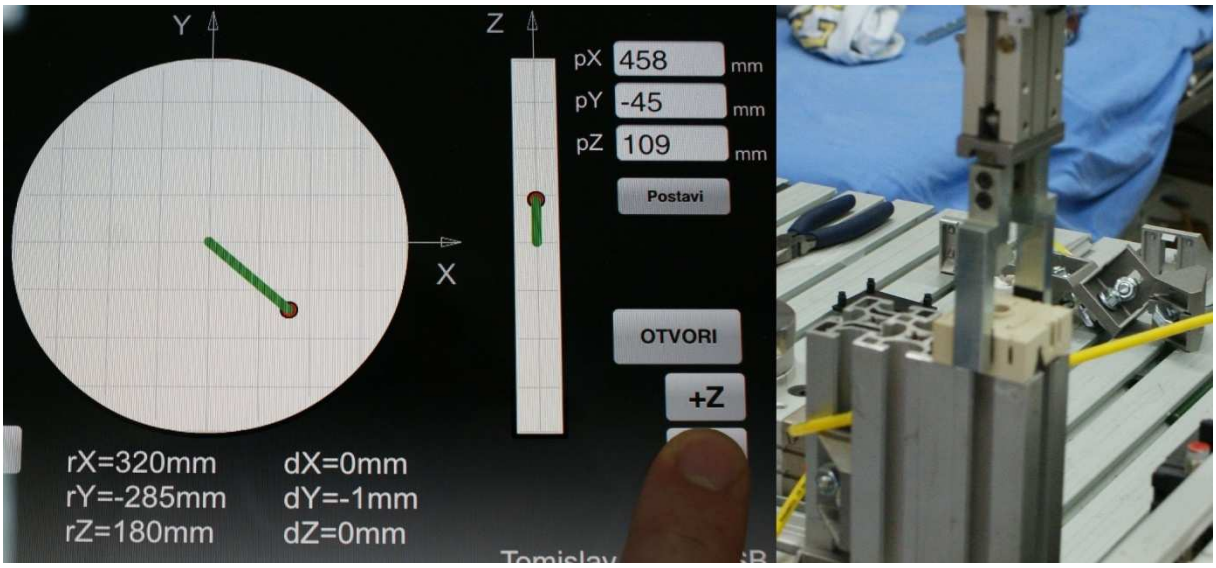
Brzinu i inkrement gibanja robota možemo mijenjati pomoću horizontalnih klizača

Slika 59. Horizontalni klizači

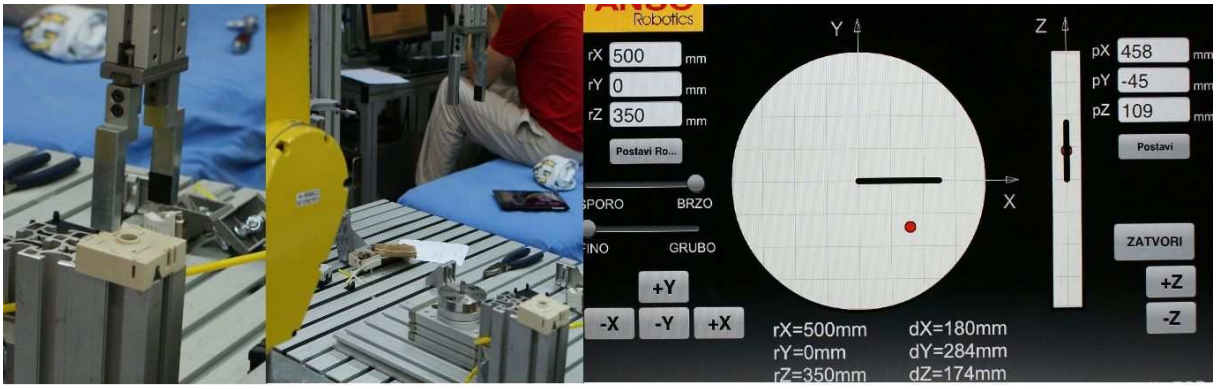
Zadaća *Pick & Place* operacije je izuzimanje predmeta od interesa s točke A i postavljanje istog u točku B. Korištenjem već objašnjenih načina upravljanja robotom zahvaćeni predmet dovodimo do željene pozicije. Otvaramo hvataljku, odmičemo je od predmeta te vraćamo robota u početnu poziciju.



Slika 60. Vođenje predmeta



Slika 61. Postavljanje predmeta



Slika 62. Odmicanje, vraćanje u početnu poziciju

7. Zaključak

Razvoj AbOvo aplikacije je bio dosta zahtjevan i iscrpljujuć posao, pogotovo za nekoga tko se je prvi put susreo s izradom aplikacija za mobilne uređaje. Ipak postignuto je upravljanje robotom pomoću iPad uređaja. Iako je upravljanje dosta ograničeno, imamo dovoljno komandi za izvođenje jednostavnih *pick & place* operacija. Povratnom vezom omogućili smo aplikaciji da na zaslonu iPad uređaja prikazuje položaj robotske ruke u prostoru. U kombinaciji s mogućnošću virtualnog postavljanja predmeta od interesa, *pick & place* operacije moguće je izvoditi bez gledanja u robotsku ruku. Dovoljno je pratiti zaslon iPad-a. Druga prednost vidljiva je u bežićnom povezivanju.

U AbOvo aplikaciji ostalo je puno prostora za daljnji razvoj i unapređenje. Prvi korak bi možda bilo pamćenje točaka te mogućnost ponavljanja naučenog programa. *Multi Touch* tehnologijom ili tri-osnim akcelerometrom moglo bi se izvesti upravljanje robotskom jedinicom na savim drugaćiji način te bi se na tome trebalo poraditi.

Ako ste novi u pisanju KAREL programa imate osjećaj da komunikacija sa iPad uređajem neće nikada proraditi. Pošto su to sve već isprobane tehnologije i imate se na što referirati, kasnije sve postane jasno i logično.

Ipad je izuzetno kvalitetan i primamljiv uređaj ali u konačnici ispada neuvjerljiv i razočaravajuć, prvenstveno zbog svog operacijskog sustava koji mu dopušta pokretanje samo aplikacija i programa odobrenih od Apple-a i distribuiranih putem Apple App Store-a. Uređaj je moguće otključati te na njemu imati neovlaštene aplikacije i programe, ali problem je u izradi vlastitih. Naime za izradu vlastith aplikacije koje će biti pokretane na iOS platformi poželjno je imati Mac računalo, ne i nužno. Aplikacije se mogu raditi i pod Windowsima, ali da bi dobili gotov proizvod moramo imati certifikat i odobrenje od Apple-a koje je potrebno platiti. S druge strane za izradu Android aplikacija nisu potrebne nikakve licence ni certifikati.

Flash Builder 4.6 program je vrlo kvalitetan, zanimljiv i prijateljski nastrojen prema korisniku. Dostupan je studentima besplatno za nekomercijalne svrhe. Idealan je za učenje. Uz veliku količinu literature i video tutoriala osnove programiranja se vrlo brzo savladaju. Potrebna nam je samo upornost i dobra ideja.

8. Prilog

8.1. Programski kod AbOvo aplikacije

8.1.1. AbOvo.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:TabbedViewNavigatorApplication xmlns:fx="http://ns.adobe.com/mxml/2009"

xmlns:s="library://ns.adobe.com/flex/spark"
                                                    applicationDPI="160"

splashScreenImage="@Embed('file:///C:/Documents and
Settings/ramelov/Desktop/ZAVRSNI RAD/AbOvoFinalVersion/New
Folder/back.jpg')"

splashScreenScaleMode="stretch">

      <s:ViewNavigator label="Pocetna" width="100%" height="100%"
firstView="views.PocetnaView">
      </s:ViewNavigator>
      <s:ViewNavigator label="Postavke" width="100%" height="100%"
firstView="views.PostavkeView"/>
      <fx:Declarations>
            <!-- Place non-visual elements (e.g., services, value objects)
here -->
      </fx:Declarations>
               </s:TabbedViewNavigatorApplication>
```

8.1.2. PocetnaView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:View xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:supportClasses="spark.skins.mobile.supportClasses.*"
xmlns:flexpad="http://code.google.com/p/flexpad/" color="#000000"
creationComplete="init()"
title="Pocetna">
<fx:Script>
<![CDATA[
import flashx.textLayout.formats.Float;

import mx.events.FlexEvent;

import spark.events.DropDownEvent;


private var gore_comm:String ="'1''1''1''011'";
private var dolje_comm:String = "'1''1''1''012'";
private var lijevo_comm:String = "'1''1''1''013'";
private var desno_comm:String = "'1''1''1''014'";
private var plus_comm:String = "'1''1''1''015'";
private var minus_comm:String = "'1''1''1''016'";
private var posR_comm:String ="";
private var hvataljkaON_comm:String = "'1''1''1''017'";
private var hvataljkaOFF_comm:String = "'1''1''1''018'";


private static var socket:Socket = new Socket();
private var timer:Timer = new Timer(1000);

public static var pref:PreferencesManager = new PreferencesManager();


private var rX:Number = 0;
private var rY:Number = 0;
private var rZ:Number = 0;
private var X:Number = 0;
private var Y:Number = 0;
private var Z:Number = 0;


private var robotVisualization:RobotVisualization;

private var sizeX:int;
private var sizeY:int;
private var maxP:int = 1000;
private var circle:Shape = new Shape();

private function init():void
{
lijevo.autoRepeat = true;
desno.autoRepeat = true;
plus.autoRepeat = true;
minus.autoRepeat = true;
gore.autoRepeat = true;
dolje.autoRepeat = true;
```

```
timer.addEventListener(TimerEvent.TIMER, timer_tick);
timer.start();

sizeX = Capabilities.screenResolutionY;
sizeY = Capabilities.screenResolutionX;

sizeX = 200;
sizeY = 480;



robotVisualization = new RobotVisualization(mySprite, sizeX, sizeY);
robotVisualization.updateRobotRotation(rX, rY, rZ);
}


private function onSocketData(event : ProgressEvent) : void
{
var re:RegExp = /\d{1,3}\.\d|-\d{1,3}\.\d/g;


while (socket.bytesAvailable)
{
var d:String=socket.readUTFBytes(socket.bytesAvailable);



var pos:Array = d.match(re)

X = parseFloat(pos[0].toString())
Y = parseFloat(pos[1].toString())
Z = parseFloat(pos[2].toString())

rX = X/4
rY = Y/4
rZ = Z/4

robotVisualization.updateRobotRotation(rX, rY, rZ)
}
}

public function timer_tick(event:TimerEvent):void
{
connect();
}

private function connect() : void {
updateDPosition();
updateRposition();

try
{
if(!socket.connected)
{
lblStatus.text = "Not connected";

socket = new Socket();
socket.connect(String(pref.getPreference('ip')),
parseInt(String(pref.getPreference('port'))));
socket.addEventListener( ProgressEvent.SOCKET_DATA, onSocketData);
}
```

```
else
{
lblStatus.text = "Connected";
}
}
catch(e:Error)
{}
}


private function updateDPosition():void
{
var ppX:Number = (rX- robotVisualization.pX)
var ppY:Number = (rY- robotVisualization.pY)
var ppZ:Number = (rZ- robotVisualization.pZ)
dPositionX.text ="dX=" + (ppX*4).toString() + "mm";
dPositionY.text ="dY=" + (ppY*4).toString() + "mm";
dPositionZ.text ="dZ=" + (ppZ*4).toString() + "mm";
}

private function updateRposition():void
{
rPositionX.text = "rX=" + X.toString() + "mm";
rPositionY.text = "rY=" + Y.toString() + "mm";
rPositionZ.text = "rZ=" + Z.toString() + "mm";
}

protected function butPostaviR_clickHandler(event:MouseEvent):void
{
var posRX:Number = parseInt(tbrX.text);
var posRY:Number = parseInt(tbrY.text);
var posRZ:Number = parseInt(tbrZ.text);
posR_comm = "'" + posRX.toString() + "'" + "'" + posRY.toString() + "'" +
"'" +  posRZ.toString() + "'" + "'111'";

connect();
socket.writeUTF(posR_comm)

}

protected function gore_buttonDownHandler(event:FlexEvent):void
{
connect();
socket.writeUTF(gore_comm);


robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function lijevo_buttonDownHandler(event:FlexEvent):void
{
connect();
socket.writeUTF(lijevo_comm);

robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function desno_buttonDownHandler(event:FlexEvent):void
{
connect();
socket.writeUTF(desno_comm);
```

```
robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function dolje_buttonDownHandler(event:FlexEvent):void
{
connect();
socket.writeUTF(dolje_comm);

robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function minus_buttonDownHandler(event:FlexEvent):void
{
connect();
socket.writeUTF(minus_comm);

robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function plus_buttonDownHandler(event:FlexEvent):void
{
connect();
socket.writeUTF(plus_comm);

robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function butPostavi_clickHandler(event:MouseEvent):void
{
var   pppX:Number = parseInt(tbX.text);
var   pppY:Number = parseInt(tbY.text);
var   pppZ:Number = parseInt(tbZ.text);
robotVisualization.pX = (pppX/4)
robotVisualization.pY = (pppY/4)
robotVisualization.pZ = (pppZ/4)

robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function grufin_changeHandler(event:Event):void
{

var brzina_comm:String = "'1''1''1'" + grufin.value.toString() + "'";

connect();
socket.writeUTF(brzina_comm);
}

protected function prhvONOFF_clickHandler(event:MouseEvent):void
{
if (prhvONOFF.label != "OTVORI")
{
prhvONOFF.label="OTVORI"
connect();
socket.writeUTF(hvataljkaON_comm);
}

else if (prhvONOFF.label != "ZATVORI")
{
prhvONOFF.label = "ZATVORI"
```

```
connect();
socket.writeUTF(hvataljkaOFF_comm)
}

if(!robotVisualization.isGrabbed)
robotVisualization.tryGrab(rX,rY,rZ);

else

robotVisualization.isGrabbed = false;

robotVisualization.updateRobotRotation(rX, rY, rZ);
}

protected function fingru_changeHandler(event:Event):void
{
var step:Number = 120 + fingru.value
var step_comm:String = "'1''1''1'" + step.toString( ) + "'"
connect();
socket.writeUTF(step_comm);
}

]]>
</fx:Script>
<fx:Declarations>
<!-- Place non-visual elements (e.g., services, value objects) here -->
</fx:Declarations>

<!--  <s:Rect id="backgroundRect" left="0" right="0" top="0" bottom="0"  >
<s:fill>
<s:LinearGradient rotation="30">
<s:GradientEntry color="#70b2ee" />
<s:GradientEntry color="#042884" />
</s:LinearGradient>
</s:fill>
</s:Rect>-->

<s:Image id="pozadina" width="100%" height="100%" chromeColor="#5F5252"
scaleMode="stretch"
source="assets/pozadina3(1).png"/>
<s:Label x="331" y="10" width="415" height="28" color="#FAF5F5"
text=" Zavod za robotiku i automatizaciju proizvodnih sustava"/>
<s:Button id="gore" x="103" y="444" width="80" height="50" label="+Y"
buttonDown="gore_buttonDownHandler(event)" chromeColor="#B8B8B8"
color="#030303"
fontSize="30"/>
<s:Button id="lijevo" x="17" y="500" width="80" height="50" label="-X"
buttonDown="lijevo_buttonDownHandler(event)" chromeColor="#B8B8B8"
color="#030303"
fontSize="30"/>
<s:Button id="desno" x="189" y="500" width="80" height="50" label="+X"
buttonDown="desno_buttonDownHandler(event)" chromeColor="#B8B8B8"
color="#030303"
fontSize="30"/>
<s:Button id="dolje" x="103" y="500" width="80" height="50" label="-Y"
buttonDown="dolje_buttonDownHandler(event)" chromeColor="#B8B8B8"
color="#030303"
fontSize="30"/>
<s:Button id="minus" x="897" y="500" width="80" height="50" label="-Z"
buttonDown="minus_buttonDownHandler(event)" chromeColor="#B8B8B8"
color="#030303"
```

```
fontSize="30"/>
<s:Button id="plus" x="897" y="444" width="80" height="50" label="+Z"
buttonDown="plus_buttonDownHandler(event)" chromeColor="#B8B8B8"
color="#030303"
fontSize="30"/>
<s:Image id="fanucind" x="0" y="-5" width="208" height="90"
source="assets/FANUC_USA_logo_small.jpg"/>
<s:Image id="fsb" x="211" y="-1"
source="assets/fsb_logo.png"/>
<s:TextArea id="tbX" x="855" y="91" width="115" height="40" color="#2C2B2B"
contentBackgroundColor="#FFFFFF" fontSize="30"/>
<s:TextArea id="tbY" x="855" y="137" width="115" height="40"
color="#2C2B2B"
contentBackgroundColor="#FFFFFF" fontSize="30"/>
<s:TextArea id="tbZ" x="855" y="183" width="115" height="40"
color="#2C2B2B"
contentBackgroundColor="#FFFFFF" fontSize="30"/>
<s:Label x="815" y="101" width="40" height="23" color="#F8F5F5"
fontSize="24" text="pX"/>
<s:Label x="815" y="147" width="34" color="#FBF7F7" fontSize="24"
text="pY"/>
<s:Label x="815" y="193" width="34" color="#FCF9F9" fontSize="24"
text="pZ"/>
<s:Label id="lblStatus" x="835" y="8" color="#FAF5F5" fontSize="20"
text="Not connected"/>
<s:Label id="lblRe" x="753" y="33" color="#F9F2F2" fontSize="20" text=""/>

<s:SpriteVisualElement id="mySprite" x="349" y="69" width="345"
height="269"/>
<s:Label id="eto" x="734" y="624" color="#FAF7F7" fontSize="29"
text="Tomislav Radic FSB"/>
<s:Button x="855" y="240" width="114" height="40" label="Postavi"
chromeColor="#B8B8B8"
click="butPostavi_clickHandler(event)" color="#030303"/>
<s:Label id="dPositionX" x="525" y="525" width="204" height="35"
color="#FFFFFF" fontSize="30"
text="0"/>
<s:Label id="rPositionX" x="313" y="525" width="204" height="35"
color="#FFFFFF" fontSize="30"
text="0"/>
<s:Label id="dPositionY" x="525" y="560" width="204" height="35"
color="#FFFFFF" fontSize="30"
text="0"/>
<s:Label id="dPositionZ" x="525" y="595" width="204" height="35"
color="#FFFFFF" fontSize="30"
text="0"/>
<s:Label x="674" y="331" color="#CCCCCC" fontSize="30" text="X"/>
<s:Label x="407" y="63" color="#CCCCCC" fontSize="30" text="Y"/>
<s:Label x="727" y="61" color="#CCCCCC" fontSize="30" text="Z"/>
<s:Label x="27" y="101" width="33" height="26" color="#FFFFFF"
fontSize="24" text="rX"/>
<s:Label x="27" y="147" width="39" color="#FFFFFF" fontSize="24"
text="rY"/>
<s:Label x="27" y="193" color="#FFFFFF" fontSize="24" text="rZ"/>
<s:Label id="rPositionY" x="313" y="560" width="204" height="35"
color="#FFFFFF" fontSize="30"
text="0"/>
<s:Label id="rPositionZ" x="313" y="595" width="204" height="35"
color="#FFFFFF" fontSize="30"
text="0"/>
<s:Label x="752" y="8" color="#FFFFFF" fontSize="20" text="STATUS:"/>
```

```
<s:TextArea id="tbrX" x="60" y="91" width="115" height="40" color="#2C2B2B"
contentBackgroundColor="#FFFFFF" fontSize="30"/>
<s:TextArea id="tbrY" x="60" y="137" width="115" height="40"
color="#2C2B2B"
contentBackgroundColor="#FFFFFF" fontSize="30"/>
<s:TextArea id="tbrZ" x="60" y="183" width="115" height="40"
color="#2C2B2B"
contentBackgroundColor="#FFFFFF" fontSize="30"/>
<s:Button x="60" y="240" width="115" height="40" label="Postavi Robota"
chromeColor="#B8B8B8"
click="butPostaviR_clickHandler(event)" color="#030303"/>
<s:HSlider id="grufin" x="10" y="296" width="200"
change="grufin_changeHandler(event)"
chromeColor="#9C9C9C" maximum="500" minimum="100" stepSize="100"
value="100"/>
<s:Label x="10" y="333" width="233" height="27" color="#FFFFFF"
fontSize="20" text="SPORO                BRZO"/>
<s:Button id="prhvONOFF" x="847" y="378" width="128" height="58"
label="ZATVORI"
chromeColor="#B8B8B8" click="prhvONOFF_clickHandler(event)" color="#030303"
fontSize="20" fontStyle="normal" fontWeight="bold" lineThrough="false"
textDecoration="none"/>
<s:HSlider id="fingru" x="10" y="364" width="200"
change="fingru_changeHandler(event)"
chromeColor="#9C9C9C" maximum="10" minimum="1" stepSize="3" value="1"/>
<s:Label x="10" y="401" color="#FFFFFF" fontSize="20" text="FINO
GRUBO"/>
<s:Label x="180" y="110" color="#FFFFFF" fontSize="20" text="mm"/>
<s:Label x="180" y="158" color="#FFFFFF" fontSize="20" text="mm"/>
<s:Label x="180" y="206" color="#FFFFFF" fontSize="20" text="mm"/>
<s:Label x="975" y="110" color="#FFFFFF" fontSize="20" text="mm"/>
<s:Label x="975" y="158" color="#FFFFFF" fontSize="20" text="mm"/>
<s:Label x="975" y="206" color="#FFFFFF" fontSize="20" text="mm"/>


</s:View>
```

8.1.3.PostavkeView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:View xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark" creationComplete="init()"
title="Postavke">

<fx:Script>
<![CDATA[

private function init():void {
txtIP.text = PocetnaView.pref.getPreference('ip').toString();
txtPort.text = PocetnaView.pref.getPreference('port').toString();
}

protected function spec_clickHandler(event:MouseEvent):void{
navigator.pushView(SpecView);
}

protected function butConnect_clickHandler(event:MouseEvent):void
{

PocetnaView.pref.setPreference('ip', txtIP.text);
PocetnaView.pref.setPreference('port', txtPort.text);
PocetnaView.pref.savePreferences();
}


]]>
</fx:Script>

<fx:Declarations>
<!-- Place non-visual elements (e.g., services, value objects) here -->
</fx:Declarations>
<s:Image id="pozadina2"  width="100%" height="100%" scaleMode="stretch"
source="assets/pozadina3.png"/>
<s:Image x="-27" y="229" source="assets/pro_r1_c5-47-TR.png"/>
<s:Button id="spec" x="910" y="597" width="90" height="45" label="specifi"
click="spec_clickHandler(event)"/>
<s:Label  x="70" y="28" color="#FAF4F4" fontSize="20" text="IP adresa:"/>
<s:TextInput id="txtIP" x="52" y="50" text="192.168.1.250"/>
<s:Label  x="69" y="104" color="#FAF3F3" fontSize="20" text="Port:"/>
<s:TextInput id="txtPort" x="52" y="126"  text="5555"/>
<s:Button id="butConnect" x="374" y="126" width="100" height="38"
label="Connect"
click="butConnect_clickHandler(event)"/>
              </s:View>
```

8.1.4. SpecView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<s:View xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"  title="SpecView">

<fx:Script>
<![CDATA[
protected function butOk_clickHandler(event:MouseEvent):void
{
navigator.popView();
}

]]>
</fx:Script>

<fx:Declarations>
<!-- Place non-visual elements (e.g., services, value objects) here -->
</fx:Declarations>
<s:Image x="-1" y="-8" source="assets/lrmate200ic-5l.jpg"/>
<s:Image x="605" y="-20" width="537" height="638"
source="assets/specifi.png"/>
<s:Button id="butOk" x="910" y="597" width="90" height="45" label="OK"
click="butOk_clickHandler(event)"/>
               </s:View>
```

8.1.5.PreferenceManager.as

```
package
{
	import flash.filesystem.File;
	import flash.filesystem.FileMode;
	import flash.filesystem.FileStream;
	import flash.utils.ByteArray;
	import flash.utils.Dictionary;
	import flash.utils.IDataInput;
	import flash.utils.IDataOutput;
	import flash.utils.IExternalizable;

	public class PreferencesManager
	{
		private var _prefValues:Dictionary;
		private var _prefsStream:FileStream;

		public function PreferencesManager()
		{
			_prefValues = new Dictionary();
			loadPreferences();
		}

		public function loadPreferences():void {

			var fp: File = File.applicationStorageDirectory;

			fp = fp.resolvePath( 'prefs.xml' );

			if ( fp.exists ) {

				_prefsStream = new FileStream();
				_prefsStream.open( fp, FileMode.READ);

				readExternal( _prefsStream );
				_prefsStream.close();

			} else {
				_prefValues[ 'ip' ] = '192.168.1.250';
				_prefValues[ 'port' ] = '5555';
				savePreferences();
			}

		}

		public function savePreferences():void {
			var fp: File = File.applicationStorageDirectory;

			fp = fp.resolvePath( 'prefs.xml' );

			_prefsStream = new FileStream();
			_prefsStream.open( fp, FileMode.WRITE );

			writeExternal( _prefsStream );
			_prefsStream.close();
		}

		public function setPreference( name:String, value:Object ):void
```

```
{
	_prefValues[ name ] = value;
	savePreferences();
}

public function getPreference( name:String ):Object
{
	return _prefValues[ name ];
}

public function readExternal(input:IDataInput):void
{
	var bytes:ByteArray = new ByteArray();
	input.readBytes( bytes, 0, input.bytesAvailable );

	var q:Object = bytes.readObject();

	_prefValues = new Dictionary();

	for( var p:Object in q ) {
		_prefValues[ p ] = q[p];
	}
}

public function writeExternal(output:IDataOutput):void
{
	var bytes:ByteArray = new ByteArray();
	bytes.writeObject( _prefValues );

	output.writeBytes( bytes );
}
}
}
```

8.1.6. RobotVisualization.as

```
package
{
	import flash.display.Shape;
	import flash.display.Sprite;
	import flash.utils.flash_proxy;

	import mx.utils.ColorUtil;

	import spark.core.SpriteVisualElement;

	public class RobotVisualization
	{
		private var lineTop:Shape = new Shape();
		private var lineSide:Shape = new Shape();
		private var circle:Shape = new Shape();
		private var circleZ:Shape = new Shape();
		private var circleOC:Shape = new Shape()

		private var sizeX:int;
		private var sizeY:int;

		public var pX:int = 0;
		public var pY:int = 0;
		public var pZ:int = 0;

		public var isGrabbed:Boolean = false;

		public function RobotVisualization(sprite:SpriteVisualElement,
sizeX:int, sizeY:int)
		{
			this.sizeX = sizeX;
			this.sizeY = sizeY;

			circleZ.graphics.lineStyle(2, 0x000000);
//predmetZ
			circleZ.graphics.beginFill(0xE70D0D);
			circleZ.graphics.drawCircle(sizeX/2 + 325, sizeY/2, 8);
			circleZ.graphics.endFill();

			circle.graphics.lineStyle(2, 0x000000);
//predmetXY
			circle.graphics.beginFill(0xE70D0D);
			circle.graphics.drawCircle(sizeX/2, sizeY/2, 8);
			circle.graphics.endFill();

			var circleBack:Shape = new Shape();
//radni prostor XY
			circleBack.graphics.lineStyle(5, 0x000000);
			circleBack.graphics.beginFill(0xfffff0);
			circleBack.graphics.drawCircle(sizeX/2, sizeY/2, 200);
			circleBack.graphics.endFill();

			var rect:Shape = new Shape();
//radni prostor Z
			rect.graphics.lineStyle(5, 0x000000);
			rect.graphics.beginFill(0xfffff0);
```

```
                rect.graphics.drawRect(sizeX/2 + 300, sizeY/2-200, 50,
400);
                rect.graphics.endFill();

                lineTop.x = sizeX/2;
                lineTop.y = sizeY/2;

                lineSide.x = sizeX/2 + 300 + 25;
                lineSide.y = sizeY/2;



                var grid:Shape = new Shape();
                grid.graphics.lineStyle(1, 0xcccccc, 100);
                grid.graphics.moveTo(sizeX/2,sizeY/2);
//X
                grid.graphics.lineTo(sizeX/2+250,sizeY/2);
                grid.graphics.lineTo(sizeX/2+230,sizeY/2+5);
                grid.graphics.lineTo(sizeX/2+230,sizeY/2-5);
                grid.graphics.lineTo(sizeX/2+250, sizeY/2);

                grid.graphics.moveTo(sizeX/2,sizeY/2);
//Y
                grid.graphics.lineTo(sizeX/2,sizeY/2-250);
                grid.graphics.lineTo(sizeX/2+5,sizeY/2-230);
                grid.graphics.lineTo(sizeX/2-5,sizeY/2-230);
                grid.graphics.lineTo(sizeX/2,sizeY/2-250);

                grid.graphics.moveTo(sizeX/2+325,sizeY/2+195);
//Z
                grid.graphics.lineTo(sizeX/2+325,sizeY/2-250);
                grid.graphics.lineTo(sizeX/2+330,sizeY/2-230);
                grid.graphics.lineTo(sizeX/2+320,sizeY/2-230);
                grid.graphics.lineTo(sizeX/2+325,sizeY/2-250);

                grid.graphics.moveTo(sizeX/2-195,sizeY/2);
//gridX
                grid.graphics.lineTo(sizeX/2,sizeY/2);
                grid.graphics.moveTo(sizeX/2-185,sizeY/2-50);
                grid.graphics.lineTo(sizeX/2+185,sizeY/2-50);
                grid.graphics.moveTo(sizeX/2-160,sizeY/2-100);
                grid.graphics.lineTo(sizeX/2+160,sizeY/2-100);
                grid.graphics.moveTo(sizeX/2-130,sizeY/2-150);
                grid.graphics.lineTo(sizeX/2+130,sizeY/2-150);
                grid.graphics.moveTo(sizeX/2-20,sizeY/2-195);
                grid.graphics.lineTo(sizeX/2+20,sizeY/2-195);
                grid.graphics.moveTo(sizeX/2-185,sizeY/2+50);
                grid.graphics.lineTo(sizeX/2+185,sizeY/2+50);
                grid.graphics.moveTo(sizeX/2-160,sizeY/2+100);
                grid.graphics.lineTo(sizeX/2+160,sizeY/2+100);
                grid.graphics.moveTo(sizeX/2-128,sizeY/2+150);
                grid.graphics.lineTo(sizeX/2+128,sizeY/2+150);
                grid.graphics.moveTo(sizeX/2-20,sizeY/2+195);
                grid.graphics.lineTo(sizeX/2+20,sizeY/2+195);

                grid.graphics.moveTo(sizeX/2,sizeY/2+195);
//gridY
                grid.graphics.lineTo(sizeX/2,sizeY/2);
                grid.graphics.moveTo(sizeX/2-50,sizeY/2-185);
                grid.graphics.lineTo(sizeX/2-50,sizeY/2+185);
```

```
                grid.graphics.moveTo(sizeX/2-100,sizeY/2-160);
                grid.graphics.lineTo(sizeX/2-100,sizeY/2+160);
                grid.graphics.moveTo(sizeX/2-150,sizeY/2-128);
                grid.graphics.lineTo(sizeX/2-150,sizeY/2+128);
                grid.graphics.moveTo(sizeX/2-195,sizeY/2-20);
                grid.graphics.lineTo(sizeX/2-195,sizeY/2+20);
                grid.graphics.moveTo(sizeX/2+50,sizeY/2-185);
                grid.graphics.lineTo(sizeX/2+50,sizeY/2+185);
                grid.graphics.moveTo(sizeX/2+100,sizeY/2-160);
                grid.graphics.lineTo(sizeX/2+100,sizeY/2+160);
                grid.graphics.moveTo(sizeX/2+150,sizeY/2-128);
                grid.graphics.lineTo(sizeX/2+150,sizeY/2+128);
                grid.graphics.moveTo(sizeX/2+195,sizeY/2-20);
                grid.graphics.lineTo(sizeX/2+195,sizeY/2+20);

                grid.graphics.moveTo(sizeX/2+305,sizeY/2);
//gridZ
                grid.graphics.lineTo(sizeX/2+345,sizeY/2);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2-50);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2-50);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2-100);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2-100);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2-150);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2-150);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2-195);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2-195);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2+50);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2+50);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2+100);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2+100);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2+150);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2+150);
                grid.graphics.moveTo(sizeX/2+305,sizeY/2+195);
                grid.graphics.lineTo(sizeX/2+345,sizeY/2+195);

                sprite.addChild(circleBack);
                sprite.addChild(rect);
                sprite.addChild(grid);
                sprite.addChild(circle);
                sprite.addChild(circleZ);
                sprite.addChild(lineTop);
                sprite.addChild(lineSide);



                updateRobotRotation(0, 0, 0);
        }

    public function updateRobotRotation(rX:int, rY:int, rZ:int):void
        {
            if(isGrabbed)
            {
                pX = rX;
                pY = rY;
                pZ = rZ;
            }


            rX = saturateVal(rX, -180, 180);
            rY = saturateVal(rY, -180, 180);
            rZ = saturateVal(rZ, -180, 180);
```

```
                pX = saturateVal(pX, -180, 180);
                pY = saturateVal(pY, -180, 180);
                pZ = saturateVal(pZ, -180, 180);

                lineTop.graphics.clear();
                lineSide.graphics.clear();

                lineSide.graphics.lineStyle(10, 0x000000);

                lineTop.graphics.lineStyle(10, 0x000000);
                if(isGrabbed)
                {
                      lineTop.graphics.lineStyle(10, 0x0D970A);
                      lineSide.graphics.lineStyle(10, 0x0D970A);
                }


                lineTop.graphics.lineTo(rX,-rY);
                lineSide.graphics.lineTo(0,-rZ);

                circle.x = pX;
                circle.y = -pY;
                circleZ.y = -pZ;

          }

          public function tryGrab(rX:int, rY:int, rZ:int):void
          {
                if(Math.abs(rX - pX) < 3 && Math.abs(rY - pY) < 3 &&
Math.abs(rZ - pZ) < 3)
                {
                      isGrabbed = true;

                }
          }


          public function saturateVal(value:int, min:int, max:int):int
          {
                if(value > max)
                      value = max;
                else if(value < min)
                      value = min;

                return value;
          }

          }
          }
```

8.2. Programski kod Karel programa

8.2.1. Ao_ipad09.kl

```
PROGRAM ao_ipad09
VAR
I, STATUS,entry,port,portC,n_bytes:INTEGER
file_var_s:FILE
s,xs,ys,zs,s1,comm,robot,reply,reply1,reply2,reply3,koor,srot:STRING[25]
REAL_VALUE:REAL
config_var:CONFIG
x1,y1,z1,w1,p1,r1,gf:REAL
P01:XYZWPR

 ROUTINE OPEN_FILE_ (FILE_ : FILE; TAG_ : STRING) FROM LIB_FILE
 ROUTINE CLOSE_FILE_(FILE_ : FILE; TAG_ : STRING) FROM LIB_FILE

BEGIN

  $GROUP[1].$UFRAME = $MNUFRAME[1,2]
  $GROUP[1].$UTOOL=$MNUTOOL[1,2]
  $GROUP[1].$MOTYPE=LINEAR;
  $GROUP[1].$SPEED=200;
  $GROUP[1].$TERMTYPE=NODECEL;

   --inicijalizacija varijabli

    x1=0
    y1=0
    z1=0

    w1=0
    p1=0
    r1=0
    I=0
    gf=1

   RDO[1]=OFF

   DELAY 10 ;

   CNV_STR_CONF('nut000',config_var,STATUS)
   x1=500;y1=0;z1=350;w1=180;p1=0;r1=+90;
   P01= POS(x1,y1,z1,w1,p1,r1, config_var)
   MOVE TO P01 NOWAIT

   GET_VAR(entry,'*SYSTEM*', '$HOSTNAME',s,STATUS)
   IF s='ROBOT1' THEN;robot='R1_';port=10257;ENDIF;
         IF s='ROBOT1' THEN;robot='R1_';portC=10258;ENDIF;
         IF s='ROBOT2' THEN;robot='R2_';port=10267;ENDIF;
         IF s='ROBOT2' THEN;robot='R2_';portC=10268;ENDIF;

  SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$OPER',0,STATUS) ;
  SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$STATE',0,STATUS) ;
```

```
  DELAY 20

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$COMMENT','SOUND',S
TATUS) ;

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$PROTOCOL','SM',STA
TUS) ;

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$REPERRS','FALSE',S
TATUS) ;

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$TIMEOUT',9999,STAT
US) ;

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$SERVER_PORT',port,
STATUS)

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$PATH','192.168.123
.103',STATUS)

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$STRT_PATH','192.16
8.123.103',STATUS)

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$REMOTE','192.168.1
23.103',STATUS)

 SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$STRT_REMOTE','192.
168.123.103',STATUS);

      DELAY 10 ;


SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$OPER',3,STATUS);

SET_VAR(entry,'*SYSTEM*','$HOSTS_CFG[6].$STATE',3,STATUS) ;

      DELAY 10 ;


SET_VAR(entry,'*SYSTEM*','$HOSTC_CFG[6].$OPER',3,STATUS);

SET_VAR(entry,'*SYSTEM*','$HOSTC_CFG[6].$STATE',3,STATUS) ;

      DELAY 10 ;

   CLOSE_FILE_(file_var_s,'S6:')

   OPEN_FILE_(file_var_s,'S6:')

      DELAY 200

      WHILE TRUE DO

        reply=''
        reply1=''
        reply2=''
        reply3=''
        koor=''

        DELAY 100;
```

```
BYTES_AHEAD(file_var_s,n_bytes,STATUS)

        IF n_bytes=0 THEN; GOTO normal_;
             ENDIF;

        WHILE TRUE DO


          BYTES_AHEAD(file_var_s,n_bytes,STATUS)
          IF n_bytes>0 THEN;

          READ file_var_s(srot::2,xs::0::2, ys::0::2,
zs::0::2,comm::0::2);ELSE; GOTO kraj_;
          ENDIF


          WRITE('NAREDBA:',comm,CR)


        ENDWHILE

        GOTO normal_
           kraj_::

             CNV_STR_INT(comm,I)

              SELECT I OF
              CASE (011): --NAPRIJED
               P01.y=P01.y+gf

              CASE (012): --NAZAD
               WRITE('nazad',CR)
               P01.y=P01.y-gf

              CASE (013): --LIJEVO
               WRITE('lijevo',CR)
               P01.X=P01.X-gf

              CASE (014): --desno
               WRITE('desno',CR)
               P01.X=P01.X+gf

               CASE(015): --gore
               WRITE('gore',CR)
               P01.z=P01.z+gf

               CASE(016): --dolje
               WRITE('dolje',CR)
               P01.z=P01.z-gf

               CASE(017): --prihvatnicaON
               RDO[1]=ON

               CASE(018): --prihvatnicaOFF
               RDO[1]=OFF

               CASE(100):
               $GROUP[1].$SPEED=50;

               CASE(200):
               $GROUP[1].$SPEED=100;
```

```
                CASE(300):
                $GROUP[1].$SPEED=200;

                CASE(400):
                $GROUP[1].$SPEED=400;

                CASE(500):
                $GROUP[1].$SPEED=600;

                CASE(121):
                gf=1;

                CASE(124):
                gf=4;

                CASE(127):
                gf=7;

                CASE(130):
                gf=10;

                CASE(111):

                CNV_STR_REAL(xs, x1);
                CNV_STR_REAL(ys, y1);
                CNV_STR_REAL(zs, z1);
                P01.x=x1;
                P01.y=y1;
                P01.z=z1;

            ENDSELECT

            MOVE TO P01

             CNV_REAL_STR(P01.x,6,1,reply1)
             CNV_REAL_STR(P01.y,6,1,reply2)
             CNV_REAL_STR(P01.z,6,1,reply3)
             koor=reply1+reply2+reply3
             WRITE file_var_s(koor,CR)

        normal_::

    ENDWHILE

    CLOSE_FILE_(file_var_s,'S6:')

END ao_ipad09
```

LITERATURA

[1] Roger Braunstein, Mims H. Wright, Joshua J. Noble: Actionscript 3.0 Bible, Wiley Publishing, Inc., 2008.

[2] The official training workbook from Adobe Systems:ActionScript 3.0 for Adobe Flash Professional CS5 Classroom in a Book, 2010.

[3] R30iA Manual CRO

[4] B-82724EN-1_01_R-30iA_LRMate_HandlingTool

[5] R-30iA_KAREL_Reference_Manual_[Ver.7.30][MARRCRLRF04071E_REV.B]

[6] R-J3iC_Internet_Options_Setup&Operations_Manual_[7.20][MAROCINOP08051E_REV.B]

[7] WEB: http://www.youtube.com/user/iBrent/videos